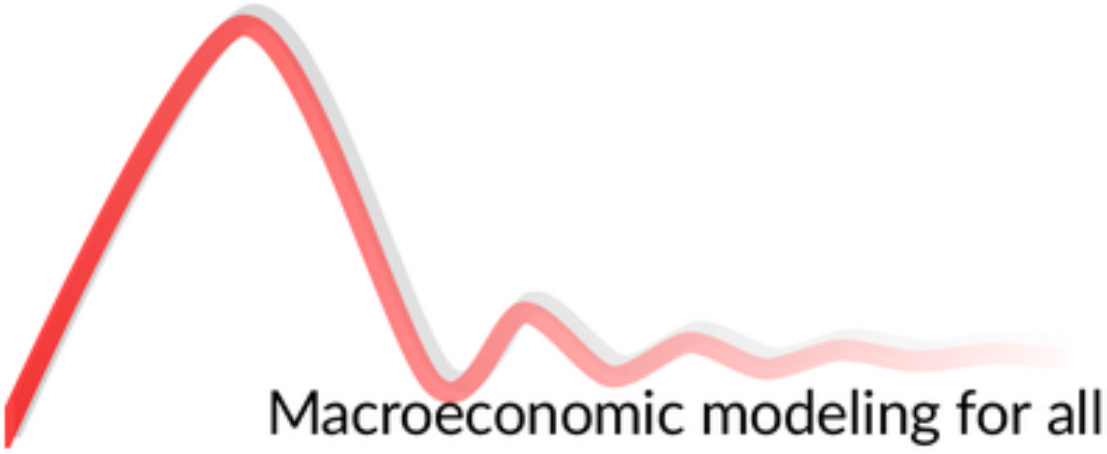

Dynare 5.1 参考手册（中文版）

Dynare Team 著

宏观研学会 译校

2022-05

版权声明

译者声明

为什么要翻译 Dynare 参考手册？

2015 年 3 月以来，我们对宏观经济学产生浓厚兴趣，经常在一起讨论、交流宏观经济现象、研究、模型与软件编程。许文立在本科期间攻读数学专业时学习过 Matlab 平台，博士阶段自学过 Sargent 的 Python、Julia 平台，而许坤则一直使用 R 语言。平心而论，Dynare 并不强大，它只是基于 Matlab 的一个预处理程序包而已。但是对于宏观经济模型（DSGEs、VARs）的定量模拟和估计来说，Dynare 非常简单易用。因此，目前大部分的宏观经济研究者基本都是用它搞点小模型，得到一点结果分析分析还是比较省事。

以前在学习 Dynare 时，并没有觉得英文版手册有多难。本来认为，做宏观经济学研究的人读英文应该是再平常不过的事。但是，自从开通“宏观经济研学会（CIMERS）”微信公众号推广宏观经济与政策的定量分析，我们发现情况并非如此，国内大部分学生刚接触 DSGE 和 Dynare 时，英文还是非常大的阻碍。因此，为了宏观经济定量分析在大中华地区更广泛地传播与应用，我们将最新版的 Dynare 手册翻译成中文，让大家看起来更容易一些。

译校团队成员

主编：许文立 安徽大学

副主编：王芝清 中国人民大学

统稿：王芝清 中国人民大学　王子君 中国人民大学

翻译：夏堰龙 北京物资学院　黄昱程 北京大学　杨雨佳 清华大学　黄聃 南华大学
温瑞昌 中山大学　常清 中央财经大学　唐蕊 中央党校（国家行政学院）
张兆洋 中国人民大学　王芝清 中国人民大学　吴峰宇 安徽大学

校对：杨瑶 中国人民大学　胡星 香港大学　郑小琴 华侨大学　陈开璞 南开大学
李铭乐 南开大学

免责声明

联系方式

如果你发现译文有错误或不当之处，望不吝赐教。

我们的电子邮件：xuweny87@163.com

致 谢

（1） Netlogo 手册的译者张发直接触动了我们启动该翻译项目。

（2） 感谢我们的父母，还有阳阳和星星。

（3） 感谢安徽大学经济学院 17 级卓越班学生的父母，因为学生们从父母那里遗传了吃苦的基因和聪明的脑袋瓜子。

（4） 感谢宏观研学会 Reading Group 团队提供的大力支持。

宏观研学会（CIMER）

2022 年 5 月

目录

1 引言 1
1.1 什么是 Dynare? 1
1.2 文档来源 2
1.3 研究引用 Dynare 的格式 2
2 安装配置 3
2.1 软件要求 3
2.2 Dynare 的安装 3
2.2.1 Windows 系统 3
2.2.2 GUN/Linux 系统 3
2.2.3 macOS 系统 4
2.2.4 FreeBSD 4
2.2.5 其他系统 4
2.3 安装编译器 5
2.3.1 Windows 系统的前提条件 5
2.3.2 GUN/Linux 系统的前提条件 5
2.3.3 macOS 系统的前提条件 5
2.4 配置 5
2.4.1 MATLAB 5
2.4.2 Octave 6
2.4.3 一些警告 6
3 运行 Dynare 7
3.1 Dynare 调用 7
3.2 Dynare 的附加程序 13
3.3 理解预处理程序的错误信息 14
4 模型文件 15
4.1 惯例 15
4.2 变量声明 16
4.2.1 动态模型变量声明 20
4.3 表达式 21
4.3.1 参数和变量参数 22
4.3.2 运算符 22
4.3.3 函数 23
4.3.4 随机模型中的几点警告 25
4.4 参数初始化 25
4.5 模型声明 26
4.6 辅助变量 30
4.7 初始条件和终止条件 31
4.8 外生变量冲击 49
4.9 其他一般性声明 54
4.10 稳态 54
4.10.1 用 Dynare 非线性算法解稳态 54
4.10.2 向 Dynare 提供稳态 58

4.10.3 稳态计算替换一些方程 60
4.11 获取模型信息 61
4.12 确定性模拟 62
4.13 随机解和模拟 66
4.13.1 计算随机解 66
4.13.2 变量类型和顺序 74
4.13.3 一阶近似 75
4.13.4 二阶近似 76
4.13.5 三阶近似 76
4.13.6 高阶近似 77
4.14 偶然受限约束（OCCBIN） 77
4.15 似然估计方法 82
4.16 矩估计方法 114
4.17 模型比较 121
4.18 冲击分解 122
4.19 校准平滑 130
4.20 预测 131
4.21 最优策略 139
4.21.1 承诺下的最优策略（Ramsey） 141
4.21.2 自由裁量权下的最优策略 143
4.21.3 最优简单规则（OSR） 143
4.22 敏感度和识别分析 147
4.22.1 执行敏感性分析 147
4.22.2 IRF/矩校准 150
4.22.3 执行识别分析 151
4.22.4 分析和输出文件类型 154
4.23 马尔科夫转换 SBVAR 160
4.24 尾声变量 168
4.25 半结构模型+ 169
4.25.1 辅助模型 169
4.25.2 VAR 预期 172
4.25.3 PAC 方程 174
4.25.4 PAC 方程估计 177
4.26 显示和保存结果 178
4.27 宏处理语言 179
4.27.1 宏表达式 179
4.27.2 宏指令 183
4.27.3 典型用法 188
4.27.4 循环与宏处理器循环 190
4.28 逐句包含 191
4.29 Misc 命令 191
5 配置文件 194
5.1 Dynare 配置 194
5.2 平行配置 195

5.3 Windows 分步指南 198
6 时间序列 200
6.1 日期 200
6.1.1 mod 文件中的日期 200
6.1.2 日期类 201
6.2 dseries 类 215
6.3 X-13 ARIMA-SEATS 接口 252
6.4 其余项目 255
6.4.1 时间聚合 255
6.4.2 利用单变量模型创建时间序列 256
7 报告 258
8 示例 269
9 Dynare 的 misc 命令 270
10 参考文献 273
附表：字段汇总表 279

1 引言

1.1 什么是 Dynare?

Dynare 是一款处理多类经济模型的软件平台，特别是动态随机一般均衡（DSGE）模型和世代交叠（OLG）模型。Dynare 求解依赖于理性预期假设的模型，即代理人的预期形成方式与模型一致。除此之外，Dynare 也能够处理不同预期形成方式的模型：代理人对未来具有完美预期，或者不完全了解经济状况，即有限理性因而在学习过程中形成预期。Dynare 求解模型涉及的代理人类型包括消费者、生产型企业、政府、货币当局、投资者和金融中介机构。在上述每种代理类别中纳入一些差异化的特征，可以分析一定程度的异质性。

Dynare 提供了一种易用直观的方式来描述这些模型。它能够根据给定模型参数校准值模拟模型，根据给定数据集估计参数。实践中，用户将编写一个包含模型变量列表、连接各变量的动态方程、执行的计算任务以及所需的图形或数值输出结果的文本文件。

Dynare 内部使用了大量的应用数学和计算机编程知识：多元非线性求解和优化、矩阵分解、局部函数近似、卡尔曼滤波和平滑、用于贝叶斯估计的蒙特卡洛马尔科夫链（MCMC）技术、图形算法、最优控制等。

各大公共机构（中央银行、经济和财政部门、国际组织）和一些私人金融机构都运用 Dynare 分析政策，并作为预测工作的工具。学术界使用 Dynare 开展研究生阶段的宏观经济学研究和教学。

Dynare 是一款免费的软件，这意味着可以免费下载、免费获得源代码，并且可以同时用于非盈利和盈利目的。大部分的源文件都被第 3 版或者更高版本的 GUN 通用公共许可证（GPL）覆盖（有一些内容除外，参见 Dynare 发布的 License 文件）。Dynare 适用于 Windows、macOS 以及 Linux 平台，在用户指南和参考手册中都有完整说明。Dynare 的一部分代码是用 C++编写的，其余部分是用 MATLAB 编程语言编写的。后者意味着运行 Dynare 需要商业付费的 MATLAB 软件。然而，作为 MATLAB 的替代品，Dynare 也可以运行 GNU Octave（基本上是 MATLAB 的免费克隆版）：对于那些不能支付或不愿支付 MATLAB，并且能够承受一定性能损失的学生或者机构来说，使用 Octave 的概率非常高。

Dynare 主要是由法国经济学研究与应用中心（Centre pour la Recherche Économique et ses Applications，CEPREMAP）的一个核心研究团队研发的，这些研究人员将部分时间用于软件研发。目前 Dynare 的研发成员有 St ephane Adjemian（Universite du Maine，Gains 和 CEPREMAP）、Houtan Bastani（CEPREMAP），Michel Juillard（Banque de France），Frederic Karame（Universite du Maine，Gains 和 CEPREMAP），Junior Maih（挪威银行），Ferhat Mihoubi（Universite Paris-Est Creteil，Epee 和 CEPREMAP），George Perendia、Johannes Pfeifer（科隆大学），Marco Ratto（欧盟委员会联合研究中心，Joint Research Centre，JRC）和 Sebastien Villemot（CEPREMAP）。与此同时，研发人员越来越多，因为 CEPREMAP 外部的研究人员开发的工具逐步整合进 Dynare。研发资金由 CEPREMAP、法国银行

和 DSGE-net（为 DSGE 建模的国际研究网络）。

Dynare 的研发人员和用户之间的互动是项目的核心。用户可以在网上论坛发帖提出问题或者上报故障。目前已知和常见的故障列举在 Dynare 百科，问题或者期望可以在我们的 Git repository 报告。培训课程的形式为一年一次的 Dynare 暑期学校，每次约有 40 人参加。最后，团队对于 Dynare 未来发展以及扩展特性的优先顺序与资金提供方合作决定。

1.2 文档来源

本文档是 Dynare 的参考手册，系统地说明了所有命令和特性，其他有用的信息来源包括 Dynare 百科和 Dynare 论坛。

1.3 研究引用 Dynare 的格式

如果您想在研究文章中引用 Dynare，推荐的参考文献格式如下：

Stéphane Adjemian, Houtan Bastani, Michel Juillard, Frédéric Karamé, Ferhat Mihoubi, Will i Mutschler, Johannes Pfeifer, Marco Ratto, Normann Rion and Sébastien Villemot (2022), "Dynar e: Reference Manual, Version 5," Dynare Working Papers, 72, CEPREMAP.

为了方便快捷，您可以复制粘贴如下信息到 BibTeX 文件：

```
@TechReport{Adjemianetal2022,
  author      = {Adjemian, St\'ephane and Bastani, Houtan and
                 Juillard, Michel and Karam\'e, Fr\'ederic and
                 Mihoubi, Ferhat and Mutschler, Willi
                 and Pfeifer, Johannes and Ratto, Marco and
                 Rion, Normann and Villemot, S\'ebastien},
  Title       = {Dynare: Reference Manual Version 5},
  Year        = {2022},
  institution = {CEPREMAP},
  type        = {Dynare Working Papers},
  number      = {72},
}
```

如果您想提供 URL，请使用 Dynare 网站的地址：http://www.dynare.org.

2 安装配置

2.1 软件要求

Dynare 软件包适用于 Windows（8.1，10 和 11）、部分 GUN/Linux 发行版（Debian、Ubuntu、Linux Mint、Arch Linux）、macOS（12“Monterey”）和 FreeBSD。Dynare 应该也可以在其他系统上运行，但是需要一些编译步骤。

为了运行 Dynare，您需要安装以下软件：

- MATLAB：介于 8.3（R2014a）到 9.12（R2022a）之间的任何版本；
- GUN Octave：介于 5.2.0 到 6.4.0 之间的任何版本，带有来自 Octave-forge 的统计包。但是注意，Windows 版 Dynare 安装程序需要一个更具体的 Octave 版本，如下载页面所示。

下面的可选扩展也可以获得额外的特性，但非必选的：

- MATLAB：优化工具箱、统计工具箱、控制系统工具箱；
- GUN Octave：下列 Octave-forge 安装包：`optim`、`io`、`statistics`、`control`。

2.2 Dynare 的安装

安装后，Dynare 可以在计算机上的任何路径中使用。最好的做法是模型文件与包含 Dynare 工具箱不在同一路径中。这样您就可以升级 Dynare，放弃以前的版本，却避免丢失模型文件。

2.2.1 Windows 系统

执行名为 `dynare-x.y-win.exe` 的自动安装程序（`x.y` 为版本号），并按照指示说明操作。默认安装路径是 `c:\dynare\x.y`。

安装后，路径将包括若干子路径，其中有 `matlab`、`mex` 和 `doc`。

安装程序还将在开始菜单中添加一个条目，并提供文档文件和卸载程序的快捷方式。

注意，您可以有多版本 Dynare（例如在 `c:\dynare` 中），只要您正确地调整了路径设置（参见 2.4.3 一些警告）。

还要注意，在命令行中传递 `/S` 标志给安装程序，也可以进行静默安装。在机房中执行无人值守的 Dynare 安装时，这是非常有用的。

2.2.2 GUN/Linux 系统

在 Debian、Ubuntu 和 Linux Mint 上，Dynare 包可以用 `apt install Dynare` 安装。这将提供可以和 Octave 同步使用的 Dynare 安装包。如果您已经安装了 MATLAB，还应该这样做：`apt install dynare-MATLAB`（Debian 环境下，安装包在 `contrib` 部分）。文档可以用 `apt install dynare-doc` 安装。安装包的状态可以在这些页面检查：

- Package status in Debian
- Package status in Ubuntu

- Package status in Linux Mint

Arch Linux 环境下，Dynare 包不在官方库中，但在 Arch User Repository 可用，从 package status in Arch Linux 下载所需的源代码。

Dynare 将安装在`/usr/lib/dynare`，文档位于`/usr/share/doc/dynare-doc`（仅在 Debian、Ubuntu 和 Linux Mint 上）。

2.2.3 macOS 系统

2.2.3.1 Matlab

为了在 Matlab 上安装 Dynare，执行名为`dynare-x.y.pkg`的自动安装程序（`x.y`为版本号），按照指示说明操作。默认安装路径是`/Applications/Dynare/x.y`，安装后，路径将包含若干子路径，其中有`matlab`、`mex`和`doc`。注意，不同版本的 Dynare 可以共存（默认情况下在`/Applications/Dynare`），只要您正确地调整了路径设置（参见 2.4.3 一些警告）。

默认情况下，安装程序会在文件夹`*.brew`下安装一个 GCC 版本（用于 *`use_dll`*）。为此，它还在同一个文件夹中安装了一个 Homebrew 版本，在一个系统文件夹中安装了 Xcode Command Line Tools（这是苹果的产品）。

所有这些都需要一点时间和硬盘空间。花费的时间将取决于计算能力和互联网连接。为了减少安装时间，您可以自己安装 Xcode Command Line Tools（参见 2.3.3 macOS 系统的前提条件）。Dynare、Homebrew 和 GCC 占大约 600MB 的磁盘空间，而 Xcode Command Line Tools 需要大约 400MB。如果您不使用 *`use_dll`* 选项，您可以选择放弃安装 GCC，Dynare 仅占用大约 50MB 的磁盘空间。

2.2.3.2 Octave

我们没有为 macOS 提供支持 Octave 的 Dynare 软件包，但在 Homebrew 中有一个支持 Octave 的 Dynare 软件包。

安装好 Homebrew 后，运行终端，通过输入以下内容安装 Dynare（和 Octave）：

```
brew install dynare
```

然后，在同一终端中运行以下程序打开 Octave：

```
octave --gui
```

最后，在 Octave 提示下，安装一些附加组件（您只需做一次）：

```
octave:1> pkg install -forge io statistics control struct optim
```

2.2.4 FreeBSD

Dynare 提供了一个 FreeBSD 端口。它可以通过以下方式安装。

```
pkg install dynare
```

2.2.5 其他系统

您需要从 Dynare 网站下载 Dynare 源代码并解压到某个位置，然后需要重新编译预处理器和动态可加载库。请参考 README.md。

2.3 安装编译器

2.3.1 Windows 系统的前提条件

Windows 没有先决条件。Dynare 现在提供的编译环境，可与 *use_dll* 选项使用。

2.3.2 GUN/Linux 系统的前提条件

Linux 环境下的 MATLAB 用户需要安装一个工作编译环境。在 Debian、Ubuntu 或 Linux Mint 环境下，可以通过 `apt install build-essential` 安装。

Linux 环境下的 Octave 用户应该安装 MEX 文件编译包（在 Debian、Ubuntu 或 Linux Mint 环境下，可以通过 `apt install liboctave-dev` 实现）。

2.3.3 macOS 系统的前提条件

2.3.3.1 Matlab

Dynare 现在提供了一个可以与 `use_dll` 选项一起使用的编译环境。要正确安装这个环境，Dynare 安装程序要确保 Xcode Command Line Tools（苹果产品）已经安装在系统文件夹中。您要自行安装 Xcode Command Line Tools，只需在终端（`/Application/Utilities/Terminal.app`）提示符中输入 `xcode-select--install` 即可。

2.3.3.2 Octave

该编译器可以通过 Homebrew 安装。在终端中运行：

```
brew install gcc-11
```

2.4 配置

2.4.1 MATLAB

您需要将 Dynare 的 `matlab` 子文件夹添加到 MATLAB 路径中，有两种做法：

- 在 MATLAB 命令窗口中使用 `addpath` 命令

在 Windows 系统环境，假设您已经在标准位置安装了 Dynare，并设置了 `x.y` 的正确的版本，输入：

```
>>addpath c:/dynare/x.y/matlab
```

在 GUN/Linux 系统环境，输入：

```
>>addpath /usr/lib/dynare/matlab
```

在 macOS 系统环境，假设您已经在标准位置安装了 Dynare，并设置了 `x.y` 的正确版本，输入：

```
>>addpath /Applications/Dynare/x.y/matlab
```

MATLAB 在下次运行时不会记住这个设置，届时您还要重新设置一遍。

- 点击菜单项

在“文件/File”菜单中选择“设置路径/Set Path”，然后单击“添加文件夹/Add Folder…”，并选择“Dynare”的 `matlab` 子路径。注意，您不应该使用“添加子文件夹/Add w

ith Subfolders…”。点击“保存/Save”应用路径设置。注意，在 MATLAB 下次运行时会记住该设置。

2.4.2 Octave

您需要使用 Octave 命令提示符上的 `addpath` 将 Dynare 安装的 `matlab` 子文件夹添加到 Octave 路径。

在 Windows 系统环境，假设您已经在标准位置安装了 Dynare，并设置了 `x.y` 的正确版本，输入：

```
octave:1>addpath c:/dynare/x.y/matlab
```

在 Debian、Ubuntu 或 Linux Mint 系统环境，不需要使用 `addpath` 命令，安装包可以执行。在 Arch Linux 系统环境，输入：

```
octave:1>addpath/usr/lib/dynare/matlab
```

在 macOS 系统，假设您已经通过 Homebrew 安装了 Dynare，输入：

```
octave:1>addpath/usr/local/lib/dynare/matlab
```

如果您不想在每次使用 Octave 时都输入这个命令，可以在主路径（Windows 系统环境通常在 `c:\Users\USERNAME`，macOS 系统环境在`/Users/USERNAME/`）里把代码放在一个名为`*.octaverc` 的文件中，每次启动时文件都由 Octave 运行。

2.4.3 一些警告

您应该非常小心 MATLAB 和 Octave 的路径。只需在命令窗口中输入 `path`，就可以显示内容。

该路径通常应该包括 MATLAB 或 Octave 的系统目录，以及 Dynare 安装的一些子目录。您必须手动添加 `matlab` 子目录，Dynare 将在运行时自动添加一些其他的子目录（取决于您的配置）。您必须验证目录来自拟使用的 Dynare 版本。

如果您的任何 M 文件与 Dynare 文件同名，在您的 MATLAB 或 Octave 路径中添加其他目录（除了 dynare 文件）可能会产生问题，然后您的程序将覆盖 Dynare 程序，导致 Dynare 不可用。

警告：不要将 `matlab` 文件夹的所有子目录添加到 MATLAB 或 Octave 路径。您必须让 Dynare 决定哪些子目录必须添加到 MATLAB 或 Octave 路径。否则，您可能会得到一个非最佳或不可用的 Dynare 安装结果。

3 运行 Dynare

为了给 Dynare 提供指令，用户必须编写一个模型文件（*model file*），文件扩展名必须是*.mod 或*.dyn。该文件包含模型描述和计算任务。详细介绍参见 4 模型文件。

3.1 Dynare 调用

一旦写好模型文件，就可以在 MATLAB 或 Octave 提示符中使用 dynare 命令调用 Dynare（以*.mod 的文件名作为调用文件）。

在实践中，处理模型文件分两步：第一步，模型和用户在模型文件中编写的处理命令，生成合适的 MATLAB 或 GUN Octave 指令；第二步，程序实际运行计算。以上都通过 dynare 命令自动运行。

MATLAB/Octave command:dynare FILENAME[.mod][OPTIONS…]

这个命令启动 Dynare，并执行 FILENAME.mod 中包含的指令。这个用户提供的文件包含模型和处理指令，如 4 模型文件描述的那样。下面列出的选项可以通过命令行传递，跟在*.mod 文件的名称后面，或者在*.mod 文件本身的第一行（见下面）。

dynare 首先在*.mod 文件上启动预处理程序。默认情况下（除非模型有 *use_dll* 选项），预处理器创建三个中间文件：

- +FILENAME/driver.m：包含变量声明和计算任务；
- +FILENAME/dynamic.m：包含动态模型方程。注意 Dynare 可能会引入辅助方程和变量（参见 4.6 辅助变量）。输出的是动态模型方程的残差，顺序依次为所声明的方程和动态模型方程的雅可比行列式。对于高阶近似，还提供了海塞矩阵和三阶导数。当计算动态模型的雅可比行列式时，各列内生变量的顺序存储在 M_.lead_lag_incidence 中。这个矩阵的行代表时间周期：第一行表示滞后（$t-1$）变量，第二行表示同期（t）变量，第三行表示超前（$t+1$）变量。矩阵的列以声明的顺序表示内生变量。矩阵中的 0 表示这个内生变量在这段时间内没有出现。M_.lead_lag_incidence 矩阵的值对应动态模型雅可比行列式中该变量的列。示例：第二个声明的变量为 c，M_.lead_lag_incidence（3,2）位置的值为 15。那么，雅可比行列式的第 15 列是 c(+1)的导数；
- +FILENAME/static.m：包含长期静态模型方程。注意 Dynare 可能会引入辅助方程和变量（参见 4.6 辅助变量）。输出的是静态模型方程的残差，遵循声明方程的顺序和静态方程的雅可比行列式。雅可比行列式的位置（i，j）表示第i个静态模型方程对第j个模型变量的导数，以声明的顺序表示。

这些文件可以查看了解模拟阶段报告的错误。

然后 dynare 将通过执行+FILENAME/driver.m 来运行计算任务。如果用户需要在不调用处理器（或者不调用 *dynare* 命令）的情况下重新运行计算任务，比如因为用户修

改了脚本，用户仅仅输入如下命令：

```
>>FILENAME.driver
```

这里有几条警告：在 Octave 环境下，*.mod 的文件名，不应使上面描述的生成的*.m 文件与 Octave 或 Dynare 提供的*.m 文件发生冲突。如果不遵守这条规则可能导致系统崩溃或意想不到的结果。特别地，这意味着*.mod 文件不能提供 Octave 或 Dynare 命令的名称。在 Octave 下，它还意味着*.mod 文件不能命名为 test.mod 或者 example.mod。

注意：引号注释

当传递包含空格（或者在 Octave 下是双引号）的命令行选项时，必须用单引号将整个选项（关键字和参数）括起来，如下面的示例所示。

示例

调用包含空格的选项 Dynare：

```
>>dynare <<modfile.mod>> '-DA=[i in [1,2,3] when i > 1]' 'con
ffile=C:\User\My Documents\config.txt'
```

选项

noclearall：默认情况下，dynare 将向 MATLAB（<R2015b）或 Octave 发送一个 clear all 命令，并删除所有工作区的变量和函数。这个选项指示 dynare 不要清空工作区。注意，从 Matlab 2015b 开始，dynare 只删除全局变量和持久使用变量函数，以便从 JIT（Just in Time）编译中获益。在这种情况下，该选项指示 dynare 不清除全局变量和函数。

onlyclearglobals：默认情况下，dynare 将向 MATLAB（<2015b）和 Octave 发送一个 clear all 命令，并将所有工作区变量删除。这个选项指示 dynare 只清除全局变量（即 M_、options_、oo_、estim_params_、bayestopt_、和 dataset_），将其他变量留在工作区中。

debug：指示预处理器编写一些有关扫描和解析*.mod 文件的调试信息。

notmpterms：指示预处理器在静态和动态文件省略临时术语；这通常会降低性能，但用于调试目的，因为它使静态和动态文件更具可读性。

savemacro[=FILENAME]：指示 dynare 保存宏处理后得到的中间文件（参见 4.27 宏处理语言）；保存的输出值将放在指定的文件中，如果没有指定文件则放在 FILENAME-macroexp.mod 中。有关传递包含空格的 FILENAME 参数的信息，参见引号注释。

onlymacro：指示预处理器只执行宏处理步骤，并在之后停止。这主要用于调试目的或独立于 Dynare 工具箱中的其他部分使用宏处理器。

linemacro：指定宏预处理器包含@#line 宏指令在哪个行上遇到和展开。只有*与 savemmacro* 结合使用时才有用。

onlymodel：指示预处理器只打印驱动文件中的模型信息，不打印 Dynare 命令（除

了 shocks 语句和参数初始化），因此不会执行计算任务。创建的辅助文件与其他情况下创建的文件（动态、静态文件等）相同。

nolog：指示 Dynare 不要在 FILENAME.log 中创建日志。默认值：创建日志。

output=second|third：指示预处理程序以给定的顺序输出导数。

language=matlab|julia：指示预处理程序为 MATLAB 或 Julia 编写输出。默认值：MATLAB。

params_derivs_order=0|1|2：当 *identification*、*dynare_senssitivity* 或 *estimation_cmd* 出现时，此选项用来约束导数对于预处理器计算的参数的顺序。0 表示没有导数，1 表示一阶导数，2 表示二阶导数。默认值：2。

nowarn：关闭所有警告。

notime：驱动程序结束时不打印总计算时间，也不保存到 oo_.time。

transform_unary_ops：将模型模块中的以下运算符转换为辅助变量：exp、log、log10、cos、sin、tan、acos、asin、atan、cosh、sinh、tanh、acosh、asinh、atanh、sqrt、cbrt、abs、sign、erf（未作特别说明，log 等价于 ln）。默认值：无强制转换。

json=parse|check|transform|compute：使预处理器以 JSON 格式输出*.mod 文件的一个版本到<<M_.dname>>/model/JSON/。何时创建 JSON 输出取决于传递的值。这些值表示预处理程序中的各种处理步骤。

如果传递 parse，输出值将在解析*.mod 文件之后，但在文件被检查之前写入一个名为 FILENAME.json 的文件（例如，如果模块中有未使用的外生变量，则 JSON 输出值将在预处理器退出之前创建）。

如果传递 check，输出值将在检查模型之后写入一个名为 FILENAME.json 的文件。

如果传递 transform，则转换模型的 JSON 输出（最大超前为 1，最小滞后为-1，替换期望操作符等）将被写入名为 FILENAME.json 的文件。原始的、未转换的模型将写入 FILENAME_original.json。

如果传递 compute，则在计算通过后写入输出值。此时转换后的模型写入到 FILENAME.json，原始模型写入到 FILENAME_original.json。动态和静态文件分别写入到 FILENAME_dynamic.json 和 FILENAME_static.json。

jsonstdout：不是将 json 要求的输出值写入文件，而是写入标准输出，即 MATLAB/Octave 命令窗口（和日志文件）。

onlyjson：一旦 json 请求的输出被写入，就退出处理。

jsonderivsimple：在 FILENAME_static.json 和 FILENAME_dynamic.json 中打印静态文件和动态文件的简化版本（不包括变量名和滞后信息）。

warn_uninit：显示未初始化的每个变量或参数的警告信息。参见 4.4 参数初始化或

者 *load_params_and_steady_state* 的参数初始化。参见 4.7 初始条件和终止条件，或者 *load_params_and_steady_state* 的内生和外生变量初始化。

console：激活控制台模式。除了 nodisplay 的行为，Dynare 不会使用图形等待栏进行长时间的计算。

nograph：激活 nograph 的选项（参见 *nograph*），Dynare 就不会产生任何图。

nointeractive：指示 Dynare 不要求用户输入。

nopathchange：默认情况下，如果 dynare/matlab 的目录没有置项，且 Dynare 的程序被其他工具箱中的程序覆盖，Dynare 将更改 MATLAB/Octave 的路径。如果希望 Dynare 的程序被覆盖，nopathchange 选项会发挥作用。或者该路径可以用在*.mod 文件的顶端来暂时修改路径（使用 MATLAB/Octave 的 addpath 命令）。

nopreprocessoroutput：防止 Dynare 打印通往预处理程序步骤的输出值以及预处理程序本身的输出值。

mexext=mex|mexw32|mexw64|mexmaci64|mexa64：编译与 *use_dll* 相关的输出时使用与您的平台相关的 mex 扩展。Dynare 可以自动设置，所以不需要自己设处理。

matlabroot=<<path>>：使用 *use_dll* 的 MATLAB 安装路径。Dynare 可以自动设置，所以您不需要自己设置。传递包含空格的<<path>>参数的信息，参见引号注释。

parallel[=CLUSTER_NAME]：告诉 Dynare 进行并行计算。如果传递 CLUSTER_NAME，Dynare 会使用指定的集群执行并行计算。否则，Dynare 会使用配置文件中指定的第一个集群。参见 5 配置文件中查找更多关于配置文件的信息。

conffile=FILENAME：如果它不同于默认值，就指定配置文件的位置。参见 5 配置文件，查找更多关于配置文件及其默认位置的信息。传递包含空格 FILENAME 参数的信息，参见引号注释。

parallel_slave_open_mode：在计算完成后，指示 Dynare 断开与从属节点的连接，仅在 dynare 完成程序时关闭连接。

parallel_test：不执行*.mod 文件时，测试配置文件中指定的并行设置。查找更多关于配置文件的信息参见 5 配置文件，。

parallel_use_psexec=true|false：对于 Windows 操作系统环境下的本地执行，设置 parallel_use_psexec=false 来使用 start 而不是 psexec，以便本地机器超过 32 核时正确分配亲和性。默认值：true。

-DMACRO_VARIABLE=MACRO_EXPRESSION：在命令行里定义一个宏观变量（与在模型文件中使用宏指令@#define 的效果相同，参见 4.27 宏处理语言），传递包含空格 MACRO_EXPRESSION 参数的信息，参见引号注释。注意，在命令行上传递的表达式可以引用在它之前定义的变量。

示例

调用带有命令行定义的 Dynare：

```
>>dynare <<modfile.mod>> -DA=true '-DB="A string with space"'
-DC=[1,2,3]'-DD=[i in C when i>1]'
```

-I<<path>>：定义一个路径来搜索宏处理器要包含的文件（使用@#include 指令），多重-I 标记可以传递到命令行。按照-I 标记通过的顺序来搜索路径，且使用第一个匹配文件。通过的这些标记优先于通过@#includepath 的标记。传递包含空格的<<path>>参数的信息，参见引号注释。

nostrict：当以下四个条件满足时，允许 Dynare 发布一个警告并继续处理：

1. 内生变量数目多于方程；
2. 在 initval 或 endval 中有未声明的符号；
3. 在 model 模块中发现了一个未声明的符号，这种情况被自动声明为外生；
4. 外生变量被声明，但在 model 模块中未使用。

fast：仅对模型选项 *use_dll* 有用。当再次运行相同的模型文件并且变量列表和方程式没有改变时，不要重新编译 MEX 文件。我们使用 32 位校验，储存在<model filename>/checksum 中。预处理器错过模型里的一个变化的可能性是很小的。如有疑问，请在没有 fast 选项的情况下重新运行。

minimal_workspace：指示 Dynare 不要将参数赋值写入预处理器生成的*.m 文件中的参数名称。运行由 MATLAB 施加的有工作空间大小约束的大型*.mod 文件上运行 dynare 时，这可能很有用。

compute_xrefs：指示 Dynare 计算交叉引用方程，将结果写入*.m 文件中。

stochastic：告诉 Dynare 求解的模型是随机的。如果在*.mod 文件中没有与随机模型（stoch_simul、estimation……）相关的 Dynare 命令，Dynare 默认认为求解的模型是确定性的。

exclude_eqs=<<equation_tags_to_exclude>>：告诉 Dynare 排除参数所指定的所有等式。由于一个*.mod 文件必须有与等式相同数量的内生变量，传递 *exclude_eqs* 时，将遵循排除内生变量的某些规则。如果 endogenous 标签已被设置为排除的等式，指定的变量将被排除。否则，如果被排除等式的左侧是一个只包含一个内生变量的表达式，则该变量被排除。如果这两个条件都不成立，处理过程就会因错停止。如果一个内生变量被 *exclude_eqs* 选项排除，而存在于一个没有被排除的等式中，它将转化为外生变量。

要指定排除哪些等式，您必须通过参数<<equation_tags_to_exclude>>。此参数要么获取指定要排除的等式标签列表，要么获取包含这些标签的文件名。

如果<<equation_tags_to_exclude>>是一个等式标签列表，可采取以下形式：

1. 给出一个参数，例如 exclude_eqs=[eq1,eq2]带有标签的公式[name='eq1']将被剔除。注意，如果在当前目录下有一个名为 eq1 的文件，Dynare 将尝试打开该文件并

从其中读取要排除的方程（参见下文有关 exclude_eqs 的文件名参数的信息）。还请注意，如果标签值包含一个空格，必须使用下面 2 中指定的变体，即 exclude_eqs=[eq1]；

2. 给出两个或多个参数，例如 exclude_eqs=[eq1,eq 2]，带有[name='eq1']和[name='eq 2']的方程将被排除；

3. 如果您想根据另一个标签名称（而不是默认的名称）来排除方程，可以将参数传递为——例如 exclude_eqs=[tagname=a tag]，如果要排除带有标签[tagname='a tag']的单个等式；或者 exclude_eqs=[tagname=(a tag,'a tag with a,comma')]，如果多个等式带有标签[tagname='a tag']和[tagname='a tag with a,comma']——将会被排除（注意括号，当指定一个以上的方程时需要括号）。注意，如果标签值包含一个逗号，逗号必须包含在单引号内。

如果<<equation_tags_to_exclude>>是文件名，该文件可采取如下形式之一：

1. 文件的每行一个等式，每行都代表传递给 name 标签的值。例如：

```
eq1
eq 2
```

将排除带有[name='eq1']和[name='eq 2']标签的方程式。

2. 文件每行一个等式，其中第一行之后的每行表示传递给第一行指定的标签的值：

```
tagname=
a tag
a tag with a, comma
```

排除带有标签[tagname='a tag']和[tagname='a tag with a,comma']的等式。注意，第一行必须以等号结束。

include_eqs=<<equation_tags_to_include>>：Dynare 只运行参数所指定的方程，换句话说，Dynare 将排除参数所未指定的所有方程。参数<<equation_tags_to_include>>的指定方式与 *exclude_eqs* 的参数相同。include_eqs 的功能是找出要排除的方程，然后根据 *exclude_eqs* 采取措施。

use_dll：指示预处理器创建包含模型方程和导数的动态可加载库（DLL），而不是将其写入 M 文件，这相当于使用 model 模块的 *use_dll* 选项。

nocommutativity：这个选项告诉预处理器在寻找共同的子表达式时不使用加法和乘法的换元性。因此，使用该选项时，等式在各种形式的输出中（LaTeX、JSON...）将显示为用户输入的样子（没有项或因子交换）。注意，使用这个选项可能会对预处理阶段的性能产生影响，尽管可能性很小。

这些选项可以通过在*.mod 文件的名称后面列出来传递到预处理器，也可以在*.mod 文件的第一行定义，这样可以避免在每次运行*.mod 文件时在命令行中重复输入。这一行必须是 Dynare 的单行注释（即必须以//开头），选项必须用空格分隔--+ options：和

+--。注意，+--之后的任何文本都舍弃。在命令行中，如果一个选项允许一个值，等号不能被空格包围。例如，json = compute 是不正确的，应该写成 json=compute。nopathchange 选项不能以这种方式指定，必须通过命令行传递。

输出

根据*.mod 文件中请求的计算任务，执行 dynare 命令会将包含结果的变量留在工作空间中以供进一步处理。更多的细节会在相关的计算任务下给出。M_、oo_和 options_结构会存储在位于 MODFILENAME/Output 文件夹，名为 FILENAME_results.mat 的文件里。如果它们存在，estim_params_、bayestopt_、dataset_、oo_recursive 和 estimation_info 也会储存在同一个文件中。注意，Matlab 默认的*.mat 文件上限大小为 2GB，您可以通过“首选项/Preference->常规/Genreal->MAT-文件”中启用 save -v7.3 选项来解除约束。

MATLAB/Octave variable:M_

包含有关模型的各种信息的结构。

MATLAB/Octave variable:options_

包含 Dynare 在计算过程中使用的各种选项的值结构。

MATLAB/Octave variable:oo_

包含有关多种计算结果的结构。

MATLAB/Octave variable:dataset_

包含用于估算数据的 dseries 对象。

MATLAB/Octave variable:oo_recursive_

包含那些在执行递归估计和预测时估计不同样本的模型时获得的 oo_结构的单元阵列。对于第i个观察范围内的样本获得的 oo_结构保存在第i个字段中。非估计端点的字段为空。

MATLAB/Octave variable:oo_.time

Dynare 运行的总计算时间，以秒为单位。如果使用了 *notime* 选项，则不设置此字段。

示例

从 MATLAB 或 Octave 提示调用 dynare，有或没有选项：

```
>>dynare ramst
>>dynare ramst.mod savemacro
```

另外，可以在 ramst.mod 的第一行传递这些选项：

```
//--+ options: savemacro, json=compute +--
```

然后调用 dynare，而没有在命令行上传递选项：

```
>>dynare ramst
```

3.2 Dynare 的附加程序

在 Dynare 预处理程序前后附加一些 MATLAB 脚本程序，并调用它们也是可行的。脚

本 MODFILENAME/hooks/priorprocessing.m 在调用 Dynare 的预处理器之前执行，可用于以编程方式转换将被预处理器读取的 mod 文件。调用 Dynare 预处理程序之后，执行脚本 MODFILENAME/hooks/priorprocessing.m，并且可用于在实际计算开始之前以编程方式转换 Dynare 预处理程序生成的文件。当且仅当在与模型文件 FILENAME 相同的文件夹中检测到上述脚本作为模型文件，FILENAME.mod 时，才执行 Dynare 预处理程序前后附加程序。

3.3 理解预处理程序的错误信息

如果预处理程序在执行*.mod 文件时发生错误，会提示错误信息。由于解析器也是这么运行的，所以有些时候这些错误会被误解。在这里，我们旨在揭开这些错误的神秘面纱。

预处理程序发出错误信息的形式如下：

1. ERROR:<<file.mod>>:line A,col B:<<error message>>;
2. ERROR:<<file.mod>>:line A,cols B-C:<<error message>>;
3. ERROR<<file.mod>>:line A,col B-line C,col D:<<error messag e>>。

前两个错误发生在单一命令行，错误二发生在多列，错误三发生在多行。

通常情况下，行和列的数目是精确的，直接提示您使用的错误语法。但是，由于解析器的工作方式，情况并非如此。错误的行号和列号最常见示例是缺少分号的情况，如以下示例所示：

```
varexo a, b
parameters c,...;
```

在这种情况下，解析器不知道在 varexo 指令结束后少了一个分号，直到它开始解析第二段以及碰到 parameters 命令。因为我们允许命令跨越很多列，因此，解析器在它运行到那里之前并不能知道第二行少了一个分号。一旦解析器开始解析第二行时，它意识到它遇到了一个它没有料到的关键词：参数。因此，它抛出了一个错误形式：ERROR:<<file.mod>>:line 2,cols 0-9:syntax error,unexpected PRARMETERS.这种情况下，您可以简单地在一行结束后加上一个分号，这样解析器就会继续运行。

记住，任何不违反 Dynare 语法，但同时又不被解析器识别的代码片段都被解释为 MATLAB 原生代码，这一点也很有帮助。这段代码将被直接传递给 driver 程序脚本。调查的 driver.m 文件有助于调试。当定义的变量名或参数名拼写错误，导致 Dynare 的解析器无法识别它们时，通常会出现这种问题。

4 模型文件

4.1 惯例

模型文件包括一系列命令和模块。每段命令和每个模块的元素都以分号（;）结束，而模块则以 end;结束。如果 Dynare 在行首或分号后遇到未知表达式，它将把该行的其余部分解析为本机 MATLAB 代码，即使存在更多以分号分隔的语句。为了防止含糊不清的错误消息，强烈建议始终只在每行中放入一条语句/命令，并在每个分号后开始一个新行[①]。

代码行可以逐行注释，也可以作为一个模块来注释。单行注释以//开始，在行末结束。多行注释由/*引入，并以*/结束。

示例

```
//This is a single line comment
```

```
var x;//This is a comment about x
```

```
/*This is another inline comment about alpha */alpha=0.3;
```

```
/*
This comment is spanning
two lines.
*/
```

注意，上述注释符号不应使用在本地以%引入注释的 MATLAB 代码区域中。在 verbatim 模块，参见 4.28 逐句包含，这将导致崩溃，因为//不是一个有效的 MATLAB 语句）。

大多数 Dynare 命令都有参数和一些额外选项，这些选项在命令关键词后面的括号中显示。一些选项用逗号隔开。

Dynare 命令描述应当遵循以下惯例：

- 用方括号（'[]'）表示选项参数或者选项；
- 用省略号（'...'）表示重复参数；
- 用竖线（'|'）表示互斥参数；
- 用 INTEGER 表示一个整数；
- 用 INTEGER_VECTOR 表示整数向量，由空格分开，并用方括号括起来；
- 用 DOUBLE 表示一个双精度数。以下的语法有效：1.1e3、1.1E3、1.1d3、1.1D3。在某些情况下，无限值 Inf 和-Inf 也有效。

① *.mod 文件必须具有以换行符结尾的行，这在文本编辑器中通常不可见。在 Windows 和 Unix 的系统环境上创建的文件以及在 OS X 和 macOS 上创建的文件始终符合此要求。在先前版本的 OS X Mac 上使用回车符创建的文件作为行尾字符。如果您收到 Dynare 解析错误 ERROR: <<mod file>>: line 1, cols 341-347: syntax error,...,并且*.mod 文件中有多行，知悉使用了回车符作为行尾字符。为了获得更多有用的错误消息，应将回车更改为换行符。

- 用 NUMERICAL_VECTOR 表示数字向量，由空格分离，并用方括号括起来；
- 用 EXPRESSION 表示在模型之外的数学表达（参见 4.3 表达式）；
- 用 MODEL_EXPRESSION 表示在模型之内的数学表达（参见 4.3 表达式和 4.5 模型声明）；
- 用 MACRO_EXPRESSION 指定宏处理器的表达式（参见 4.27.1 宏表达式）；
- 用 VARIABLE_NAME 表示以英文字母开头的变量名称，并且不能包含：()+-/^=!;:@#.或者突出字符；
- 用 PARAMETER_NAME 表示以英文字母开头的参数名称，并且不能包含：()+-/^=!;:@#.或者突出字符；
- 用 LATEX_NAME 表示数学模式一个有效的 LATEX 表达式；
- 用 FUNCTION_NAME 表示一个有效的 MATLAB 函数名称；
- 用 FILENAME 表示一个在底层操作系统中有效的文件名，当指定扩展名或者这个文件名包含一个不是以字母开头的参数名称时，把它放到引号里是必要的；
- 用 QUOTED_STRING 表示一个任意的字符串，包含在（单）引号之间。

4.2 变量声明

虽然 Dynare 允许用户选择自定义变量名称，但我们仍要记住一些约束：首先，变量和参数名不能和 Dynare 的命令或者内置函数的名称相同，并且 Dynare 不区分大小写。比如，不要用 `Ln` 或者 `Sigma_e` 命名变量，违背此规则可能会产生难以调试的错误消息或崩溃；然后，在运行用户定义的稳态文件时，建议避免使用 MATLAB 函数的名称，因为同样可能导致冲突，特别是当运行稳态文件时，不要使用正确拼写的希腊字母名，如 *alpha*，因为 Matlab 函数也有相同的名称，宁可选择 `alppha` 或 `alph`；最后，不要用 `i` 命名变量或者参数，否则可能会干扰虚数 i 和许多循环中的索引指标。例如，投资命名为 `invest`，不建议使用已经表示逆运算符的 `inv`。使用以下命令声明变量和参数：

```
Command:var VAR_NAME[$TEX_NAME$][(long_name=QUOTED_STR|NAME=QUOTED_STR)]...;
Command:var(deflator=MODEL_EXPR)VAR_NAME(...same options apply)
Command:var(log_deflator=MODEL_EXPR)VAR_NAME(...same options apply)
```

必选命令，声明模型内生变量。*VAR_NAME* 和 *MODEL_EXPR* 的语法参见 4.1 惯例，可以选择为变量提供 LaTeX 名称。如果变量非平稳的，还可以提供平减指数的信息。列表中的变量可以用空格或逗号分隔。`var` 命令可以在文件中出现多次，Dynare 会连接它们。Dynare 按照声明的顺序，将参数列表存储在列元胞数组 `M_.endo_names` 中。

选项

如果这个模型是非平稳的，并且写入 `model` 模块，Dynare 需要恰当的内生变量趋势

平减指数使模型平稳。趋势平减指数必须与遵循这一趋势的变量一同提供。

deflator=MODEL_EXPR：对内生变量去趋势。在 MODEL_EXP 中引用的所有趋势变量、内生变量和参数都必须用 trend_var、log_trend_var、var 和 parameters 命令声明。平减指数假定是可乘的。可加性平减指数用 log_deflator 表示。

log_deflator=MODEL_EXPR：与 deflator 相同，除了平减指数被假定为加法而非乘法（或者换句话说，声明的变量等于具有乘法趋势的变量的对数）。

long_name=QUOTED_STRING：变量名的长字符版本。值储存在 M_.endo_names_long（列元胞数组，顺序与 M_.endo_names 相同）。如果提供多个 long_name 选项，那最后一个选项也会被使用。默认值：VAR_NAME。

NAME=QUOTED_STRING：创建变量名称的分区，在*.m 文件得到直接的输出结果，类似于 M_.endo_partitions.NAME=QUOTED_STRING。

示例（可变分区）

```
var c gnp cva (country=`US',state=`VA')
          cca (country=`US',state=`CA',long_name=`Consumption C
A');
var(deflator=A) I b;
var c $C$ (long_name=`Consumption');
```

Command:varexo VARIABLE_NAME[$LATEX_NAME$][(long_name=QUOTED_STRING|NAME=QUOTE_STRING...)]...;

可选命令，声明模型外生变量。具体参见 4.1 惯例的 VAR_NAME 句法。可以选择给变量提供 LaTeX 的名称。如果用户希望对模型施加冲击，则需要声明外生变量。varexo 命令可以在文件中出现多次，Dynare 会连接它们。

选项

long_name=QUOTED_STRING：类似于 *long_name*，但是值储存在 M_.exo_names_long。

NAME=QUOTED_STRING：类似于 *partitioning*，但 QUOTED_STRING 储存在 M_.exo_partitions.NAME。

示例

```
varexo m gov;
```

备注

从某种意义上说，不能从当前经济状况预测到这个变量，外生变量就是一项创新。例如，如果记录的 TFP 是一阶自回归过程：

$$a_t = \rho a_{t-1} + \varepsilon_t$$

然后记录的 TFP，即a_t是用 var 声明的内生变量，其最佳预测是ρa_{t-1}，而技术冲击ε_t

将用 varexo 声明。

Command:varexo_det VAR_NAME[TEX_NAME][(long_name=QUOTED_STR|NAME=QUOTED_STR)...];

可选命令，声明随机模型外生确定性变量。VARIABLE_NAME 的语法参见 4.1 惯例。可以选择给变量提供 LaTeX 的名称。列表中的变量可以用空格或逗号分隔。varexo_det 命令可以在文件中出现多次，Dynare 将它们连接起来。

结合确定性冲击和随机冲击建立模型是可能的，模型中的代理人从模拟的开始就知道未来的外生变化。此时 stoch_simul 将计算添加未来信息到状态空间的理性期望解（在 stoch_simul 的输出中没有显示任何信息），而 forecast 将计算基于初始条件和未来信息的模拟。

注意，外生确定性变量不能在模型中以超前或滞后的方式出现。

选项

long_name=QUOTED_STRING: 类似于 *long_name*，但是值储存在 M_.exo_det_names_long。

NAME=QUOTED_STRING: 类似于 *partitioning*，但是 QUOTED_STRING 储存在 M_.exo_det_partitions.NAME。

示例

```
varexo m gov;
varexo_det tau;
```

Command:parameters PARAM_NAME[TEX_NAME][(long_name=QUOTED_STR|NAME=QUOTED_STR)...]

声明模型参数，或者变量初始化模块使用的参数，或者冲击声明中的参数。PARAM_NAME 的句法参见 4.1 惯例。可以选择给参数提供 LaTeX 的名称。紧随其后，这些参数必须被赋值（参见 4.4 参数初始化）。列表中的参数可以用空格或逗号分隔，parameters 命令可以在文件中出现多次，Dynare 会连接它们。

选项

long_name=QUOTED_STRING: 类似于 *long_name*，但是值储存在 M_.param_names_long。

NAME=QUOTED_STRING: 类似于 *partitioning*，但 QUOTED_STRING 储存在 M_.param_partitions.NAME。

示例

```
parameters alpha,bet;
```

Command:change_type(var|varexo|varexo_det|parameters)VAR_NAME|PARAM_NAME...;

指定变量/参数的类型更改为另一种类型：内生变量、外生变量、外生确定性变量或参数。重要的是要理解这个命令对*.mod 文件有全局影响：类型更改在 change_type 命令前后均有效。通常用于转换一些变量进行稳态校准，通常使用一个单独的模型文件校准，其中包括宏处理器的变量声明列表，并转换一些变量。

示例

```
var y,w;
parameters alpha,bet;
...
change_type(var) alpha,bet;
change_type(parameters) y,w;
```

此处的 alpha 和 beta 是内生变量，y 和 w 是参数。

Command:predetermined_variables VAR_NAME...;

Dynare 默认惯例是变量的时间反映确定此变量时间。典型的例子是资本存量：当前使用的资本存量实际上是在前一个时期决定，那么进入生产函数的资本存量是 k（-1），资本运动法则写成：

```
k=i+(1-delta)*k(-1)
```

换句话说，对于存量变量，Dynare 默认使用“期末存量”概念，而不是“期初存量”。

predetermined_variables 用来改变这个惯例。声明为前定变量的内生变量应该在所有其他内生变量的前一个时期决定。存量变量应该遵循“期初存量”惯例。

注意，即使使用 predermined_variables 命令输入模型，Dynare 内部也始终使用“期末存量”概念。因此，在绘图、计算或模拟变量时，Dynare 将遵循该约定，使用在当前期间决定的变量。例如，生成资本脉冲响应函数时，Dynare 将绘制当前投资决定的资本存量 k（将用于下一期的生产函数）的图形。这就是资本的影响有滞后性的原因，因为图中画出的是 k 而不是前定 k(-1)。特别是要记住，这也会影响来自前定变量平滑过程中模拟的时间序列和结果。与非前定变量相比，它们可能也会错误地被前移一期。

示例

以下两个程序片段是严格等价的。

使用默认 Dynare 时间惯例：

```
var y,k,i;
...
model;
y=k(-1)^alpha;
k=i+(1-delta)*k(-1);
...
```

```
end;
```

使用备选时间惯例：

```
var y,k,i;
predetermined_variables k;
...
model;
y=k^alpha;
k(+1)=i+(1-delta)*k;
...
end;
```

Command:trend_var(growth_factor=MODEL_EXPR)VAR_NAME[$LATEX_NAME$]...

可选命令，声明模型趋势变量。MODEL_EXPR 和 VAR_NAME 的语法参见 4.1 惯例。可以选择为变量提供 LaTeX 名称。假定变量具有乘法增长趋势。加法增长趋势使用 log_trend_var。

如果用户希望在 model 模块中写入非平稳模型，需要趋势变量。trend_var 命令必须出现在 var 命令之前，以供趋势变量参考。trend_var 命令可以在文件中出现多次，Dynare 会连接它们。

如果模型是非平稳的，并且写入 model 模块，Dynare 将需要每个趋势变量的增长因子来使得模型平滑。必须使用 growth_factor 关键字，在趋势变量声明中提供增长因子。MODEL_EXPR 中引用的所有内生变量和参数都必须由 var 和 parameters 命令声明。

示例

```
trend_var (growth_factor=gA) A;
```

Command:log_trend_var(log_growth_factor=MODEL_EXPR) VAR_NAME[$LATEX_NAME$]...;

与 trend_var 相同，除了该变量应该具有加法趋势（换句话说，等于具有乘法趋势的变量的对数）。

Command:model_local_variable VARIABLE_NAME[LATEX_NAME]...;

可选命令，声明模型局部变量。VARIABLE_NAME 语法参见 4.1 惯例。因为您可以在模型模块中动态创建模型局部变量（参见 4.5 模型声明），所以这个命令的主要目的是为模型局部变量分配一个 LATEX_NAME。

示例

```
model_local_variable GDP_US $GDPUS$;
```

4.2.1 动态模型变量声明

内生变量、外生变量和参数也可以在模型模块中声明。您可以通过两种不同的方式来

实现这一点：要么通过等式标签，要么直接在等式中。

在等式标签中声明动态变量，只需声明要声明的变量类型（endogenous、exogenous 或 parameters，后面加等号和单引号中的变量名）。因此，要在等式标记中声明变量 c 为内生的，可以输入[endogenous='c']。

在等式中声明动态变量，只需遵循一条垂直线的符号名（|，管道字符）和一个 e，x 或 p。例如，在模型模块中声明一个名为 alphaa 参数，可以在一个出现该变量的方程中编写 alphaa|p。类似地，在模型模块中声明一个内生变量 c，可以写成 c|e。注意，方程中的动态变量声明必须对同期变量实施。动态变量声明不必出现在遇到该变量的第一个位置。

示例

以下两个程序代码片段等价：

```
model;
  [endogenous='k',name='law of motion of capital']
  k(+1)=i|e+(1-delta|p)*k;
  y|e=k^alpha|p;
  ...
end;
delta=0.025;
alpha=0.36;
```

```
var k,i,y;
parameters delta,alpha;
delta=0.025;
alpha=0.36;
...
model;
  [name='law of motion of capital']
  k(1)=i|e+(1-delta|p)*k;
  y|e=k|e^alpha|p;
  ...
end;
```

4.3 表达式

Dynare 区分两种数学表达式：描述模型的表达式和模型模块外部使用的表达式（例如，用于初始化参数或变量，或作为命令选项）。本手册分别用 MODEL_EXPRESSION 和 EX

PRESSION 表示。

不同于与 MATLAB 或 Octave 表达式，Dynare 表达式必须是标量，不能包含矩阵或矩阵计算[②]。表达式可以使用整数（INTEGER）、浮点数字（DOUBLE）、参数名称（PARAMETER_NAME）、变量名称（VARIABLE_NAME）、运算符和函数来构造。

在某些情形中还接受了以下特殊常数：

`Constant:inf` 表示无穷。

`Constant:nan` “不是数字”：表示未定义或无法表示的值

4.3.1 参数和变量参数

只需输入参数和变量的名称，就可以在表达式中引入参数和变量。无论在模型模块内部还是外部使用，参数和变量的语义都是完全不同的。

4.3.1.1 模型内部

模型内使用的参数指的是通过参数初始化提供的值（参见 4.4 参数初始化）或在进行模拟时，`homotopy_setup` 生成，或是进行估算时的估计变量。

MODEL_EXPRESSION 使用的变量表示当期值，而不是给定的超前或滞后值。变量名跟着的括号里用整数表示超前和滞后变量：正整数表示超前期，负值表示滞后期。允许超过一期的超前或滞后。例如，如果 `c` 是一个内生变量，则 `c(+1)` 是一期超前变量，而 `c(-2)` 是两期滞后变量。

声明超前和滞后的内生变量必须遵循以下惯例：变量的时间反映了决定该变量的时间；控制变量——根据定义在当期决定——没有超前期；前定变量——根据定义在前期决定——必有滞后期。结果是所有存量变量必须使用“期末存量”惯例。

超前和滞后主要用于内生变量，但也可用于外生变量。它们对参数没有影响，对于局部模型变量是禁用的（参见 4.5 模型声明）。

4.3.1.2 模型外部

模型模块外部的表达式使用的参数或变量，只是引用了最后给定的数值。更准确地说，参数引用了相应参数初始化中给定的值（参见 4.4 参数初始化）；内生或外生变量是指在最近的 `initval` 或 `endval` 模块中给定的值。

4.3.2 运算符

MODEL_EXPRESSION 和 EXPRESSION 都允许使用以下运算符：

- 二元算术运算符：+、-、*、/、^
- 一元算术运算符：+、-
- 二元比较运算符（计算结果为 0 或 1）：<、>、<=、>=、==、!=

注意，除了在二维实平面的线上，这些运算符处处可微。然而，为了保证牛顿型方法

② 注意：所有的 MATLAB 和 Octave 表达式都可以出现在`*.mod` 文件中，但是这些表达式需要放在单独的命令行中，为了后期处理的目的，通常将其放在 mod 文件的最后。Dynare 并不编译它们，仅仅只是无修正地传递给 MATLAB 或 Octave。这些结构在本部分并不讨论。

收敛，Dynare 假设在不可微点上，这些算子对两个变量的偏导数都等于 0（因为这是其他位置偏导数的值）。

MODEL_EXPRESSION（但在 EXPRESSION 中不可以）中接受以下特殊运算符：

Operator:STEADY_STATE(MODEL_EXPRESSION)

获取封闭表达式的稳态值。典型的用法是在泰勒规则中，用稳定状态下的 GDP 值来计算产出缺口。外生变量和外生确定性变量可能不会出现在 MODEL_EXPRESSION 中。

警告：因为长期和潜在冲击，稳态的概念在完美预期的情况下是不明确的。Dynare 将使用 `oo_.steady_state` 的内容作为调用 `STEADY_STATE()`-运算符的参考。有 `endval` 情况时，将使用用户提供的终端状态。这可能是由 Dynare 计算的稳定状态（如果 `endval` 后面是 `steady`）或仅仅是用户提供的终端状态（如果 `endval` 后面没有 `steady`）。换句话说，Dynare 不会根据相应时期内外生变量的特定值自动计算出有条件的稳态。

Operator:EXPECTATION(INTEGER)(MODEL_EXPRESSION)

该运算符使用与当前可用信息不同的信息集来获取某些表达式的期望。例如，使用上个周期可用的信息集，`EXPECTATION(-1)(x(+1))`等于下个周期的变量 `x` 的期望值。内部如何处理此运算符以及如何影响输出的说明参见 4.6 辅助变量。

4.3.3 函数

4.3.3.1 内置函数

MODEL EXPRESSION 和 EXPRESSION 内部支持以下标准函数：

Function:exp(x) 自然指数

Function:log(x)

Function:ln(x) 自然对数

Function:log10(x) 常用对数

Function:sqrt(x) 平方根

Function:cbrt(x) 立方根

Function:sign(x) 符号函数定义为

$$sign(x) = \begin{cases} -1 & if\ x < 0 \\ 0 & if\ x = 0 \\ 1 & if\ x > 0 \end{cases}$$

注意，这个函数在$x = 0$处不连续，因此不可微的。然而，为了保证牛顿型方法收敛，Dynare 假设$x = 0$处的导数等于 0。这个假设来自观察到在这一点上的左导数和右导数都存在并且都等于 0，所以通过假设$x = 0$处的导数为 0 来消除奇异点。

Function:abs(x) 绝对值

注意，该连续函数在$x = 0$时不可微。但是为了保证牛顿型方法收敛，Dynare 假定$x = 0$处的导数等于 0（即使该导数不存在）。没有数学依据的合理性取决于以下观察结果：对于$\mathbb{R}$中的任何$x \neq 0$，$abs(x)$的导数等于$sign(x)$，而$sign(x)$在$x = 0$时函数值的约定。

Function:sin(x)

Function:cos(x)

Function:tan(x)

Function:asin(x)

Function:acos(x)

Function:atan(x)

以上为三角函数。

Function:max(a,b)

Function:min(a,b)

以上为两个实数的最大值和最小值。

注意，除了由$a = b$定义的二维实平面线外，这些函数处处可微。然而，为了保证牛顿型方法收敛，Dynare 假设在不可微处，函数的偏导数相对于第一个（第二个）参数等于 1（0），即扭结点的导数等于在半平面上观察到的导数，该函数等于其第一个参数。

Function:normcdf(x)

Function:normcdf(x,mu,sigma) 高斯累积密度函数，均值为 *mu* 和标准差为 *sigma*。注意 normcdf(x)与 normcdf(x,0,1)等价。

Function:normpdf(x)

Function:normpdf(x,mu,sigma)

高斯概率密度函数，均值为 *mu* 和标准差为 *sigma*。注意 normpdf(x)与 normpdf(x,0,1)等价。

Function:erf(x) 高斯误差函数。

4.3.3.2 外部函数

任何其他的自定义（或内置）MATLAB 或 Octave 函数都可以在 MODEL_EXPRESSION 和 EXPRESSION 使用，前提是此函数具有一个标量参数作为返回值。

若要在 MODEL_EXPRESSION 中使用外部函数，必须使用 external_function 语句声明该函数。而对于模型模块或 steady_state_model 模块之外的 EXPRESSION 中使用的外部函数，则不是必选的。

Command:external_function(OPTIONS...);

声明模型模块中使用的外部函数，在模型模块中使用的每个唯一函数都需要。

external_function 命令可以在文件中多次出现，且必须位于模型模块之前。

选项

name=NAME：函数名称，还必须是实现它的 M-/MEX 文件名称。此选项必选。

nargs=INTEGER：函数参数数量。如果未提供此选项，Dynare 假定 nargs=1

first_deriv_provided[=NAME]：如果提供了 NAME，则告诉 Dynare 提供了雅

可比行列式作为 M-/MEX 文件的唯一输出，作为选项参数给出。如果未提供 NAME，则告诉 Dynare，传递给 NAME 的参数指定的 M-/MEX 文件返回雅可比行列式，作为其第二个输出参数。如果未提供此选项，则 Dynare 在需要时使用有限差分近似来计算函数的导数。

second_deriv_provided[=NAME]：如果提供了 NAME，这将告诉 Dynare，海塞矩阵作为选项参数提供的 M-/MEX 文件的唯一输出。如果未提供 NAME，这将告诉 Dynare，由传递给 NAME 的参数指定的 M-/MEX 文件返回该海塞矩阵，作为其第三个输出参数。注意：只有在同一 external_function 命令中使用 first_deriv_provided 选项时，才能使用此选项。

示例

```
external_function(name=funcname);
external_function(name=otherfuncname,nargs=2,first_deriv_provided,second_deriv_provided);
external_function(name=yetotherfuncname,nargs=3,first_deriv_provided=funcname_deriv);
```

4.3.4 随机模型中的几点警告

在随机模型中，强烈建议不要使用下列函数和运算符：max、min、abs、sign、<、>、<=、>=、==、!=。原因是如果稳定状态远离扭结点，stoch_simul 或 estimation 所使用的局部近似会自动忽略这些函数引入的非线性特征。而且，如果稳定状态正好在扭结点上，那么近似将是虚假的，因为这些函数在扭结点上的导数是虚假的（如这些函数和运算符各自的文档中所述）。

注意，extended_path 不受此问题的影响，因为不依赖于局部近似的方式。

4.4 参数初始化

Dynare 模拟计算需要校准模型参数，这是通过参数初始化完成的。语法如下所示：

```
PARAMETER_NAME=EXPRESSION;
```

以下是校准示例：

```
parameters alpha,beta;


beta=0.99;
alpha=0.36;
A=1-alpha*beta;
```

参数值存储在 M_.params 内部：

MATLAB/Octave variable:M_.params

包含模型参数的值。参数顺序与 parameters 命令中使用的顺序相同，因此按 M_.param_names 中的顺序排序。

参数名称存储在 M_.param_names：

MATLAB/Octave variable:M_.param_names

包含模型参数名称的元胞数组。

MATLAB/Octave command:get_param_by_name('PARAMETER_NAME');

给定参数名称，返回存储在 M_.params 的校准值。

MATLAB/Octave command:set_param_value('PARAMETER_NAME',MATLAB_EXPRESSION);

将参数校准值设置为给定表达式。这与上述参数初始化语法基本相同，除了可以接受任意的 MATLAB/Octave 表达式，并且可以在 MATLAB/Octave 脚本运行。

4.5 模型声明

模型在 model 模块内声明：

Block:model;

Block:model(OPTIONS...);

模型的方程式写在 model 和 end 关键字限定的模块内。除了使用 ramsey_model、ramsey_policy 或 discretionary_policy 计算无约束最优策略外，方程数必须与模型内生变量数相同。

方程式的语法必须遵循 MODEL_EXPRESSION 的约定，如 4.3 表达式所述。每个方程式必须以分号（";"）结束。一个正常的方程式格式如下：

MODEL_EXPRESSION=MODEL_EXPRESSION;

当方程式以齐次形式书写时，可以省略"=0"部分，只写等式的左边，方程格式如下：

MODEL_EXPRESSION;

模型模块内，Dynare 允许创建模型局部变量，构成了若干方程共享一个表达式的简单方法。该语法由#组成，其后是新的模型局部变量名称（严禁声明为如 4.2 变量声明所列的形式，但可能已经由 *model_local_variable* 声明）、等号以及此新变量所代表的表达式。之后，每当此变量出现在模型中时，Dynare 将用分配的变量表达式替换。注意，此变量的范围仅限于模型模块，不能在模型模块外使用。为了分配 LaTeX 名字到模型局部变量，使用 *model_local_variable* 概述的声明语法。模型局部变量声明格式如下：

#VARIABLE_NAME=MODEL_EXPRESSION;

模型模块中写入方程标签也是可能的。标签可以用作不同的目的，允许用户对每个方程附加任何信息，并在运行时恢复。例如，用 name 标签命名方程式，使用如下语法：

```
model;

[name='Budget constraint'];
c+k=k^theta*A;
```

```
end;
```

此处的 name 是关键字，表示给方程式命名的标签。如果用标签命名模型方程式，resid 命令将显示方程式的名称（可能比方程式编号信息量更大）而不是方程编号。一个方程式的几个标签可以用逗号隔开。

```
model;

[name='Taylor rule',mcp='r>-1.94478']
r=rho*r(-1)+(1-rho)*(gpi*Infl+gy*YGap)+e;

end;
```

更多标签信息参见 Dynare 百科。

选项

linear：声明模型为线性。这就不需要声明初始值来计算平稳线性模型的稳态值。此选项不能与非线性模型一起使用，它不会触发模型的线性化。

use_dll：指示预处理器创建包含模型方程式和导数的动态可加载库（DLL），而不是编写在 M 文件中。您需要一个工作的编译环境，例如，一个工作的 mex 命令（参见 2.3 安装编译器了解更多详细信息）。使用此选项可使得模拟或估计更快，但是要浪费一些初始编译时间。另外，该选项也可用于 Dynare 命令（参见 3.1 Dynare 调用）③。

block：执行模型的模块分解，并在计算（稳态、确定性模拟、一阶近似随机模拟和估计）中利用它。有关确定性模拟和稳态计算中使用的算法的详细信息，参见 Dynare 百科。

bytecode：使用模型的 bytecode 表示形式，例如，包含所有方程式的紧凑形式的二进制文件，而不是 M 文件。

cutoff=DOUBLE：在模型标准化期间将雅可比行列式元素视为 null 的阈值。仅与 block 选项一起使用。默认值：1e-15。

mfs=INTEGER：控制内生变量的最小反馈集的处理，仅与 block 选项一起使用。可能的值：

0：所有内生变量都被视为反馈变量（默认值）；

1：分配给已经自动标准化方程式的内生变量（即x在y中不出现的形式$x = f(y)$)是潜在的递归变量。所有其他变量都被强制属于反馈变量集；

2：除了 mfs=1 的变量，与可以标准化的线性方程相关的内生变量是潜在的递归变量。所有其他变量都被强制属于反馈变量集；

3：除了 mfs=2 的变量，与可以标准化的非线性方程相关的内生变量是潜在的递归变

③ 尤其是对于大型模型，编译步骤非常耗时，使用该选项可能就不划算。

量。所有其他变量都被强制属于反馈变量集。

no_static：不要创建静态模型文件，这对于没有稳态的模型很有用。

differentiate_forward_vars

differentiate_forward_vars=(VARIABLE_NAME[VARIABLE_NAME...])

告诉 Dynare 为每个超前形式的内生变量创建一个新的辅助变量，以使得新的变量是原变量的一个时间差分。更确切地说，如果模型包含 `x(+1)`，则将创建一个变量 `AUX_DIFF_VAR`，使得 `AUX_DIFF_VAR=x-x(-1)`，而 `x(+1)`将被替换为 `x+AUX_DIFF_VAR(+1)`。

如果在未列出具体变量的情况下给出该选项，则将转换应用于所有超前内生变量。如果有列出，则转换将约束在列出的超前内生变量。

此选项对于某些难以收敛性确定性模拟非常有用。在非常持久的动态或永久冲击的情况下，较差的终止条件可能会妨碍正确的求解或收敛。新的微分变量具有明显的零终止条件（如果终止条件为稳态），并且在许多情况下有助于模拟收敛。

parallel_local_files=(FILENAME[,FILENAME]...)：声明在执行并行计算时应传输到从属节点的额外文件列表（参见 5.2 平行配置）。

balanced_growth_test_tol=DOUBLE：在平衡增长路径测试中用于确定交叉导数是否为零的度量（后者记录在 https://archives.dynare.org/DynareWiki/RemovingTrends）。默认值：`1e-6`。

示例（基本 RBC 模型）

```
var c k;
varexo x;
parameters aa alph bet delt gam;

model;
c=-k+aa*x*k(-1)^alph+(1-delt)*k(-1);
c^(-gam)=(aa*alph*x(+1)*k^(alph-1)+1-delt)*c(+1)^(-gam)/(1+bet);
end;
```

示例 2（使用模型局部变量）

以下程序：

```
model;
#gamma=1-1/sigma;
u1=c1^gamma/gamma;
u2=c2^gamma/gamma;
end;
```

与下列程序等价：

```
model;
u1=c1^(1-1/sigma)/(1-1/sigma);
u2=c2^(1-1/sigma)/(1-1/sigma);
end;
```

示例 3：线性模型

```
model(linear);
x=a*x(-1)+b*y(+1)+e_x;
y=d*y(-1)+e_y;
end;
```

Dynare 使用 `write_latex_original_model` 命令，将模型方程式的原始列表输出到 LaTeX 文件中，转换后的模型方程式列表使用 `write_latex_dynamic_model` 命令，静态模型方程式列表使用 `write_latex_static_model` 命令。

Command:write_latex_original_model(OPTIONS);

创建两个 LaTeX 文件：包含模型模块中定义的模型，包含 LaTeX 文档标题信息。

如果*.mod 文件是 FILENAME.mod，Dynare 创建一个名为 FILENAME/latex/original.tex 的文件，其中包括一个包含所有原始模型方程式的列表，名为 FILENAME/latex/original_content.tex 的文件（也由 Dynare 创建）。如果为变量和参数提供了 LaTeX 名称（参见 4.2 变量声明），则使用这些名字，否则将使用纯文本名称。时间下标（t、t+1、t-1、...）将附加到变量名称上，如 LaTeX 下标。

编译 TeX 文件需要以下 LaTeX 包：geometry、fullpage、breqn。

选项

write_equation_tags：在 LaTeX 输出中编写方程标签，用 LaTeX 标记。

Command:write_latex_dynamic_model;

Command:write_latex_dynamic_model(OPTIONS);

创建两个 LaTex 文件：包含动态模型，包含 LaTex 文档标题信息。

如果*.mod 文件是 FILENAME.mod，那么 Dynare 将创建一个名为 FILENAME/latex/dynamic.tex 的文件，其中包括一个包含所有动态模型方程式的列表，名为 FILENAME/latex/dynamic_content.tex 的文件（也由 Dynare 创建）。如果为变量和参数提供了 LaTeX 名称（参见 4.2 变量声明），则使用这些名字，否则使用纯文本名称。时间下标（t、t+1、t-1、...）将附加到变量名称上，如 LaTex 下标。

注意，在 TeX 文件中写入的模型将在以下几个方面不同于用户声明的模型：

- 前定变量的时间惯例（参见 *predetermined_variables*）更改为 Dynare 默认的时间惯例。换言之，声明为前定的变量将作为滞后变量；

- 删除期望运算符，由辅助变量和新方程式替换（参见 *EXPECTATION*）；
- 移除大于或等于两期的超前或滞后内生变量，替换为新的辅助变量和方程式；
- 新的辅助变量和方程替换超前或滞后外生变量。

所需的 LaTex 包的信息参见 *write_latex_original_model*。

选项

write_equation_tags：参见 *write_equation_tags*。

Command:write_latex_static_model(OPTIONS);

创建两个 LaTex 文件：包含静态模型，包含 LaTex 文档标题信息。

如果*.mod 文件是 FILENAME.mod，然后 Dynare 将创建一个名为 FILENAME/latex/static.tex 的文件，其中包括一个包含所有稳态模型方程式的列表，名为 FILENAME/latex/static_content.tex 的文件（也由 Dynare 创建）。如果为变量和参数提供了 LaTeX 名称（参见 4.2 变量声明），则将使用这些名字，否则将使用纯文本名称。

注意，在 TeX 文件中写入的模型将在以下几个方面不同于用户声明的模型（具体细节参见 *write_latex_dynamic_model*）；此命令不会输出选项 steady_state_model 模块的内容（参见 *steady_state_model*），它会输出模型模块中声明的动态模型的静态版本（即无超前和滞后）。为了写入 steady_state_mode 的 LaTeX 内容，参见 *write_latex_steady_state_model*。

LaTex 包的信息参见 *write_latex_original_model*。

选项

write_equation_tags：参见 *write_equation_tags*。

Command:write_latex_steady_state_model()

创建两个 LaTex 文件：包含稳态模型，包含 LaTex 文档标题信息。

如果*.mod 文件是 FILENAME.mod，然后 Dynare 将创建一个名为 FILENAME/latex/steady_state.tex 的文件，包括一个包含所有稳态模型方程式的列表，名为 FILENAME/latex/steady_state_content.tex 的文件（也由 Dynare 创建）。如果为变量和参数提供了 LaTeX 名称（参见 4.2 变量声明），则使用这些名字，否则使用纯文本名称。

注意，在 TeX 文件中写入的模型将在一些方面不同于用户声明的模型（具体细节参见 *write_latex_dynamic_model*）。所需的 LaTex 包的信息，参见 *write_latex_original_model*。

4.6 辅助变量

Dynare 所解的模型与用户声明的模型并非完全相同。在某些情况下，Dynare 会引入一些辅助性内生变量，连同相应的辅助方程，它们会出现在最终输出里。主要的变换涉及超前和滞后期变量。Dynare 会转换模型，使内生变量只有一期超前和一期滞后，外生变量没有超前/滞后期。这种变换是通过创建辅助变量和相应的辅助方程实现。例如，如果模型中

存在 x(+2)，Dynare 将创建一个辅助变量 AUX_ENDO_LEAD=x(+1)，并用 AUX_ENDO_LEAD(+1) 替换 x(+2)。而滞后期大于 2 的内生变量（辅助变量前缀为 AUX_ENDO_LAG）、超前和滞后的外生变量（辅助变量的前缀分别为 AUX_EXO_LEAD 或 AUX_EXO_LAG）。

另一个转换涉及 EXPECTATION 运算符，Dynare 都会创建新方程定义的辅助变量，并引用新辅助变量替换期望运算。例如，表达式 EXPECTATION(1)(x(+1)) 替换为 AUX_EXPECT_LAG_1(-1)，并且新辅助变量声明为 AUX_EXPECT_LAG_1=x(+2)。

预处理程序也在 ramsey_model 和 ramsey_policy 命令中引入辅助变量。当计算拉姆齐问题的一阶条件时，辅助变量表示拉格朗日乘数。新变量采用 MULT_i 的形式，其中i表示与乘数关联的约束（按模型模块中的声明顺序计数）。

辅助变量也可由模型模块的 differentiate_forward_vars 选项引入。新变量采用 AUX_DIFF_FWRD_i 的形式，对于某些内生变量 x 来说，辅助变量等于 x-x(-1)。

最后，在使用 diff 运算符的情况下也会出现辅助变量。

创建后，所有辅助变量都包含在内生变量集内。决策规则（见下文）的输出是这样的：辅助变量名称被它们引用的原始变量替换。

在创建辅助变量之前，内生变量的数量存储在 M_.orig_endo_nbr，而在创建辅助变量后，内生变量的数量存储在 M_.endo_nbr。

辅助变量的更多技术细节参见 Dynare 百科。

4.7 初始条件和终止条件

大多数模拟练习需要初始条件（也可能需要终止条件），还需要为非线性算法提供初始猜测值。本节介绍用于这些目的的语句。

许多情形（确定性或随机性）必须计算非线性模型的稳态，initval 就是为非线性算法声明初始值。resid 命令用于计算给定初始值下方程残差。

在完美预期模式中使用 Dynare 为前瞻性模型需要初始条件和终止条件。这些初始条件和最终条件通常是静态平衡，但也不一定。典型的应用是在第 0 时期，经济体处于均衡状态，在第一期触发冲击，然后研究回到初始均衡的轨迹。要做到这一点，需要 initval 和 shock（参见 4.8 外生变量的冲击）。另一应用是研究一个经济体从第 0 期任意初始条件开始收敛至均衡的路径。此时 histval 允许在模拟开始前为带有滞后期的变量设定不同的历史初始值。由于 Dynare 的设计结构，initval 用于声明终止条件。

Block:initval;

Block:initval(OPTIONS...);

initval 模块有两个主要用途：第一，在完美预期模拟环境下，为非线性求解提供猜测值；第二，在完美预期和随机模拟下，为稳态计算提供猜测值。依赖于 histval 和 endval 模块，它还用于在完美预期模拟练习下声明初始条件和终止条件。因为完美预期模拟下，initval 模块的意义随着 histval 和 endval 模块的存在而改变，所以我们强烈

建议检查，在运行 perfect_foresight_setup 之后和运行 perfect_foresight_solver 之前构造的 oo_.endo_simul 和 oo_.exo_simul 的变量是否包含了所需的值。存在超前和滞后期时，结果结构的子字段将滞后期的历史值存储在首行/列中，而将超前期的终值储存在末行/列。

initval 模块以 end;结束，包含下列形式的命令：

VARIABLE_NAME=EXPRESSION;

在确定性（即完美预期）模型中

首先，存储内生变量和外生变量的 oo_.endo_simul 和 oo_.exo_simul 都将用这个模块提供的值填充。如果没有其他模块，它将提供所有内生和外生变量的初始条件和终止条件，因为矩阵的末行/列也要填充。并为中间模拟期提供求解的初始值。如果存在 histval 模块（不存在 endval 模块），histval 模块将通过设置 oo_.endo_simul 和 oo_.exo_simul 的第一列/行，为状态变量提供/覆盖历史值（滞后项）。这意味着 histval 存在时，initval 模块仅设置超前变量的终值，并为求解完美预期提供初值。

由于 initval 具有各种功能，通常需要在 initval 模块为所有内生变量提供数值。滞后/超前变量严格需要初始条件和终止条件，求解器也需要可行的起始值。特别注意，如果一些内生变量或外生变量没有被 initval 模块提及，则假定为零。当使用 varexo 指定外生变量时，记住这一点特别重要，这些外生变量不允许取 0，例如 TFP。

注意，如果 initval 模块后面紧跟着 steady 命令，含义稍有改变。假设外生变量保持在 initval 模块声明的数值不变，steady 命令将计算模型所有内生变量的稳态。稳态值依赖于所声明的外生变量，写入 oo_.endo_simul，并充当历史和终止条件以及求解器的初值。因此，initval 模块跟着 steady 等价于 initval 模块加上设定外生变量的值，而内生变量的设定和外生变量的稳态值关联。

在随机性模型中

initval 主要目的是在稳态计算中为非线性求解器提供初始猜测值。注意，如果 initval 模块之后没有 steady，则稳态计算将由后续命令（stoch_simul、estimation...）触发。因此，initval 允许在为 ramsey_model 计算提供解析条件稳态文件时，指定用于寻找稳态的初始工具值，也不需要将外生随机变量的初始值声明为 0，因为它是唯一可能的值。

随后计算的稳态（不是初始值，使用 histval）将被用作随机模式下，第一个模拟期之前的所有时期，三种可能的模拟类型的初始条件：

- *stoch_simul*，如果指定 period 选项；
- *forecast* 作为预测计算的初始点；
- *conditional_forecast* 作为条件预测计算的初始点。

若要在一组特定的非计算的稳态初始值上启动模拟，请使用 *histval*。

选项

all_values_required: 如果至少有一个内生变量或外生变量未在 initval 模块中设置，则报错并停止运行*.mod 文件。

示例

```
initval;
c=1.2;
k=12;
x=1;
end;

steady;
```

Block:endval;

Block:endval(OPTIONS...);

模块以 end;结束，模块中间包含下列命令形式：

VARIABLE_NAME=EXPRESSION;

endval 模块仅在确定性模型中才有意义，不能与 histval 一起使用。与 initval 命令类似，它将模块中声明的值填充到 oo_.endo_simul 中的内生变量和 oo_.exo_simul 中的外生变量。如果没有 initval 模块，则填充整个矩阵，因此为所有内生和外生变量提供的初始条件和终止条件，因为它也将填充这些矩阵的首末行/列。由于填充了中间模拟期，它也将为求解器提供初始值。

如果存在 initval 模块，initval 将为变量提供历史值（如果存在状态变量/滞后变量），而 endval 将填充矩阵的其余部分，因此仍然提供模型中带有超前期变量的终止条件，以及针对完美预期求解器给出的所有模拟时期所有内生变量的初始猜测值。

注意，如果 endval 模块中未提到某些内生变量或外生变量，则假定相应的变量为上一个 initval 模块或 steady 命令（如果存在）中提供的值。因此，与 initval 相反，遗漏变量在并不会自动赋为 0。同样，强烈建议在运行 perfect_foresight_setup 之后和运行 perfect_foresight_solver 之前检查构造的 oo_endo_simul 和 oo_exo_simul 的变量，查看是否实现了所需的结果。

像 initval 一样，如果 endval 模块后有 steady 命令，含义稍有改变。假设外生变量保持在 endval 模块声明的数值不变，steady 命令将计算模型所有内生变量的稳态。声明的外生变量（内生变量）稳态值随后写入 oo_.endo_simul，并充当历史和终止条件以及求解器的初值。因此，endval 模块跟着 steady 等价于 endval 模块加上设定外生变量的值，而内生变量的设定和外生变量的稳态值关联。

选项

all_values_required：参见 *all_values_required*。

示例

```
var c k;
varexo x;

model;
c+k-aa*x*k(-1)^alph-(1-delt)*k(-1);
c^(-gam)-(1+bet)^(-1)*(aa*alph*x(+1)*k^(alph-1)+1-delt)*c(+1)
^(-gam);
end;

initval;
c=1.2;
k=12;
x=1;
end;

steady;

endval;
c=2;
k=20;
x=2;
end;

steady;

perfect_foresight_setup(periods=200);
perfect_foresight_solver;
```

上述例子的问题是找到$t=1$到$T=200$期间的消费和资本的最优路径，给定外生技术水平 x 的路径，c 为前向变量，外生变量 x 在实物资本预期收益中出现超前项，而 k 为纯后向（状态）变量。

初始均衡由 steady 条件下的 x=1 和终止条件下的 x=2 计算。initval 模块为 k 设置初始条件（因为它是唯一的前向变量），而 endval 模块为 c 设置终止条件（因为它是

唯一的前向内生变量)。完美预期求解器的初始值由 endval 模块给出。更多细节见下：

示例

```
var c k;
varexo x;

model;
c+k-aa*x*k(-1)^alph-(1-delt)*k(-1);
c^(-gam)-(1+bet)^(-1)*(aa*alph*x(+1)*k^(alph-1)+1-delt)*c(+1)
^(-gam);
end;

initval;
k=12;
end;

endval;
c=2;
x=1.1;
end;

perfect_foresight_setup(periods=200);
perfect_foresight_solver;
```

此例没有 steady 命令，因此条件与 initval 和 endval 模块中指定的条件完全相同。超前变量 c 和 x 需要终止条件，滞后变量 k 需要初始条件。

在没有 shocks 模块的 endval 模块中设置 x=1.1，意味着技术在$t = 1$时处于 1.1 且永远停留在那里，因为 endval 填充了 oo_.endo_simul 和 oo_.exo_simul 的所有条目，除了没有明确指定数值时，存储初始条件且 initval 模块设置为 0 的首个条目。

因为资本的运动法则是后向的，所以我们需要$t = 0$时的 k 初始条件。由于 endval 的存在，这不能通过 histval 模块来完成，必须在 initval 模块中指定。同样，由于欧拉方程是前瞻性的，我们需要在 endval 模块中指定$t = 201$处为 c 设置终止条件。

可以看出，没有必要在 initval 模块中指定 c 和 x，在 endval 块中指定 k，因为它们对结果没有影响。由于第一期的优化问题是给定$t = 0$时继承的前定资本存量 k 以及技术 x 的当前和未来值，在$t = 1$时选择 c、k，故$t = 0$时的 c 和 x 不起作用。这同样适用于$t = 200$时 c、k 的选择，它不依赖于$t = 201$时刻的 k。正如欧拉方程所示，选择仅取决于

当前资本以及未来消费 c 和技术 x，而不取决于未来资本 k。直观的原因是这些变量是分别在$t=0$和$t=201$期间发生的，但这里没有建模的优化问题结果。

示例

```
initval;
c=1.2;
k=12;
x=1;
end;

endval;
c=2;
k=20;
x=1.1;
end;
```

此例前向变量 x 和 c 有了初始条件，而后向变量 k 也有了终止条件。如前一个例子所示，这些值不会影响模拟结果。Dynare 把它们视为给定，基本假设存在外生变量和状态使这些选择成为均衡值（基本是未申明时期$t<0$和$t>201$的初始/终止条件）。

上述例子说明了另一种方法来观察在 initial 和 endval 之后 steady 的使用。与其说模拟范围前后的隐性非指定条件必须符合模块中内生变量的初始/终止条件，不如说稳态指定在$t<0$和$t>201$时的条件等同于给定 initial 和 endval 模块的外生变量时处于稳态。$t=0$和$t=201$的内生变量随后被设置为相应的稳态均衡值。

将 initval 和 endval 中指定的 c 在$t=0$和 k 在$t=201$作为给定的事实对于绘制内生变量的模拟向量有重要意义，即 oo_endo_simul 的行包含初始条件和终止条件，因此在示例中长度是 202 期。分别为前向变量和后向变量指定初始条件和终止条件的任意值，在$t=1$和$t=200$时，这些值与内生确定值相距甚远。尽管$t=0$和$t=201$的值与$0<t<201$的动态无关，也可能导致很奇怪的大跳跃。上述例子使用 *rplot* 或手工绘制 oo_.endo_val 时，消费从$t=0$到$t=1$会有大跳跃，资本从$t=200$到$t=201$亦是如此。

Block:histval;

Block:histval(OPTIONS...);

在确定性完美预期环境中

在滞后期高于一期的模型中，histval 模块允许为状态变量的不同时期声明不同的历史初始值。此时的 initval 模块发挥着设定终止条件的作用，并为求解器提供初始值。注意，histval 模块不声明非状态变量。

模块以 end;结束，并包含下列命令形式：

VARIABLE_NAME(INTEGER)=EXPRESSION;

EXPRESSION 可以返回一个数值的任何有效表达式，可以包含已经初始化的变量名。

按照 Dynare 的惯例，时期 1 是模拟的第一期。向前追溯，模拟开始前一期是时期 0，然后是时期-1 等等。假设 histval 模块未初始化的状态变量在 0 期及之前都赋值为 0。注意，histval 之后面不跟 steady。

示例

```
model;
x=1.5*x(-1)-0.6*x(-2)+epsilon;
log(c)=0.5*x+0.5*log(c(+1));
end;

histval;
x(0)=-1;
x(-1)=0.2;
end;

initval;
c=1;
x=1;
end;
```

此例的 histval 为内生变量 x 的两个滞后期设定历史条件，储存在 oo_endo_simul 的首列。initval 模块为前向变量 c 设定终止条件，储存在 oo_endo_simul 的末列。此外，initval 模块还为完美预期求解器定义了内生变量 c 和 x 的初始值。

在随机模拟环境中

随机模拟环境中，histval 允许在状态空间中设置模拟的初始点。对于完美预期模拟，所有未明确声明的变量都设置为 0。此外，由于只有状态变量进入递归策略函数，所有控制变量的声明值都将被忽略，使用情形如下：

- 在 *stoch_simul*，如果声明了 periods 选项。注意，这只影响模拟的起始点，但不影响脉冲响应函数。使用 *loglinear* 选项时，histval 模块仍然采用未取对数的起始值。
- 在 *forecast*，作为计算预测值的初始点。使用 *loglinear* 选项时，histval 模块仍然采用未取对数的起始值。
- 在 *conditional_forecast*，对于校准模型来说，作为计算条件预测的初始点。使用 *loglinear* 选项时，histval 模块仍然采用未取对数的起始值。

- 在 *Ramsey Policy* 中，指定了计算中央计划者目标函数的内生状态（包括滞后外生变量）的值。注意，无法设置与中央计划者问题关联的拉格朗日乘子初始值（参见 *evaluate_planner_objective*）。

选项

all_values_required：参见 *all_values_required*。

示例

```
var x y;
varexo e;

model;
x=y(-1)^alpha*y(-2)^(1-alpha)+e;

end;

initval;
x=1;
y=1;
e=0.5;
end;

steady;

histval;
y(0)=1.1;
y(-1)=0.9;
end;

stoch_simul(periods=100);
```

Command:resid;

将显示模型静态方程的残差，使用最后一个 `initval` 或 `endval` 模块中内生变量的值（或者用户自定义稳态文件提供的值，参见 4.10 稳态）。

Command:initval_file(OPITIONS...);

在确定性模型中为所有内生变量和外生变量声明一个路径。路径长度必须等于模拟期长度，加上模型超前期和滞后期（例如，具有 50 个模拟期，2 个滞后期和 1 个超前期，路

径长度必须为 53）。注意，这些路径囊括了两个不同条件：

- 问题的约束条件，由外生变量路径和内生变量的初值和终值给出；
- 非线性求解器的初始猜测值，由模拟期内生变量的路径（不包括初始条件和终止条件）给出。

在完美预期和随机模拟的情况下，steady 使用 initval_file 加载的首个观测值作为猜测值求解模型稳态，首个观测值在使用 first_obs 选项时由该选项决定。

不要把 initval_file 和 initval 语句混在一起。然而，initval_file 之后可以用 histval 或 histval_file 语句修改历史初始值。

模型文件中可以有几个 initval_file 语句。每条语句都会重置 oo_.initval_series。

选项

filename=FILENAME(deprecated)

datafile=FILENAME：包含数据的文件名称。如果文件名包含路径或扩展名，则必须包含在引号中。接受以下格式：

- M-file（扩展名为*.m）：对于每个内生变量和外生变量，文件必须包含相同名称的行或列向量；
- MAT 文件（扩展名为*.mat）：与 M 文件相同；
- Excel 文件（扩展名为*.xls 或*.xlsx）：对于每个内生变量和外生变量，文件必须包含一个同名列。注意：Octave 只支持*.xlsx 文件扩展名，且必须安装 io 包（键入“pkg install -forge io”即可完成）。首列可以包含每个观测值的日期；
- CSV 文件（扩展名为*.csv）：对于每个内生变量和外生变量，文件中必须包含一个同名列。首列可以包含每个观察值的日期。

first_obs={INTEGER|DATE}：文件要使用的第一个观察数据的编号或日期（参见 6.1.2 日期类）。

first_simulation_period={INTEGER|DATE}：文件的观测数据或模拟（或预测）开始的日期（参见 *dates*）。这个选项可以避免计算模型中的最大滞后数。对应于首个模拟期的观测值不需要存在于文件中，因为初始化所需的唯一日期在该日期之前。

last_obs={INTEGER|DATE}：文件中要使用的最后一个观测数据的编号或日期（参见 6.1.2 日期类）。

nobs=INTEGER：文件中要使用的观测值数量（从 first_obs 的首个观测值开始）。

series=DSERIES NAME：包含数据的 DSERIES 的名称（参见 6.2 dseries 类）。

示例 1

```
var c x;
varexo e;
```

```
parameters a b c d;

a=1.5;
b=-0,6;
c=0.5;
d=0.5;

model;
x=a*x(-1)+b*x(-2)+e;
log(c)=c*x+d*log(c(+1));
end;

initval_file(datafile=mydata.csv);

perfect_foresight_setup(periods=200);
perfect_foresight_solver;
```

初始值和终端值取自文件 mydata.csv（不能保证这些值是模型稳态）。轨迹的猜测值也取自该文件。该文件必须至少包含 203 个变量 c、x 和 e 的观测值。如果文件中的观测值超过 203 个，则 perfect_foresight_setup(period=200)将使用前 203 个。注意，与 x(-2)对应的辅助变量值是 initval_file 自动计算的。

示例 2

```
var c x;
varexo e;
parameters a b c d;

a=1.5;
b=-0,6;
c=0.5;
d=0.5;

model;
x=a*x(-1)+b*x(-2)+e;
log(c)=c*x+d*log(c(+1));
end;
```

```
initval_file(datafile=mydata.csv,first_obs=10);

perfect_foresight_setup(periods=200);
perfect_foresight_solver;
```

初始值和终端值取自文件 mydata.csv，从第 10 个观测值开始。文件中至少要有 212 个观测值。

示例 3

```
var c x;
varexo e;
parameters a b c d;

a=1.5;
b=-0,6;
c=0.5;
d=0.5;

model;
x=a*x(-1)+b*x(-2)+e;
log(c)=c*x+d*log(c(+1));
end;

ds=dseries(mydata.csv);
lds=log(ds);

initval_file(series=lds,first_obs=2010Q1);

perfect_foresight_setup(periods=200);
perfect_foresight_solver;
```

初始值和终端值取自 dseries 的 lds。所有观测值都从 2010 年第一季度开始加载，直到文件结束，必须包含至少到 2050Q3 的数据。

示例 4

```
var c x;
varexo e;
```

```
parameters a b c d;

a=1.5;
b=-0,6;
c=0.5;
d=0.5;

model;
x=a*x(-1)+b*x(-2)+e;
log(c)=c*x+d*log(c(+1));
end;

initval_file(datafile=mydata.csv,first_simulation_period=2010
Q1);

perfect_foresight_setup(periods=200);
perfect_foresight_solver;
```

初始值和终端值取自文件 mydata.csv。文件中的观测值必须有日期。所有的观测值都是从 2009 年第三季度开始加载，直到文件结束。文件中必须有至少到 2050Q1 的数据。

示例 5

```
var c x;
varexo e;
parameters a b c d;

a=1.5;
b=-0,6;
c=0.5;
d=0.5;

model;
x=a*x(-1)+b*x(-2)+e;
log(c)=c*x+d*log(c(+1));
end;
```

```
initval_file(datafile=mydata.csv,last_obs=212);


perfect_foresight_setup(periods=200);
perfect_foresight_solver;
```

初始值和终端值取自文件 mydata.csv。加载前 212 个观测值，perfect_foresight_setup(period=200)使用前 203 个观测值。

示例 6

```
var c x;
varexo e;
parameters a b c d;


a=1.5;
b=-0,6;
c=0.5;
d=0.5;


model;
x=a*x(-1)+b*x(-2)+e;
log(c)=c*x+d*log(c(+1));
end;


initval_file(datafile=mydata.csv,first_obs=10,nobs=203);


perfect_foresight_setup(periods=200);
perfect_foresight_solver;
```

初始值和终端值取自文件 mydata.csv。加载第 10 至 212 观测数据。

示例 7

```
var c x;
varexo e;
parameters a b c d;


a=1.5;
b=-0,6;
c=0.5;
```

```
d=0.5;


model;
x=a*x(-1)+b*x(-2)+e;
log(c)=c*x+d*log(c(+1));
end;


initval_file(datafile=mydata.csv,first_obs=10);


steady;
```

mydata.csv 的第 10 个观测值用作猜测值计算稳态。外生变量设置为文件中的数值，如果这些变量不存在，则设置为零。

Command:histval_file(OPTIONS...);

和 histval 等价，只是从输入文件读取，通常与 smoother2histval 一起使用。

选项

filename=FILENAME(deprecated)

datafile=FILENAME：包含数据的文件名。接受以下文件格式：

- M-file（扩展名为*.m）：对于每个内生变量和外生变量，文件必须包含相同名称的行或列向量；
- MAT 文件（扩展名为*.mat）：与 M 文件相同；
- Excel 文件（扩展名为*.xls 或*.xlsx）：对于每个内生变量和外生变量，文件必须包含一个同名列。注意：Octave 只支持*.xlsx 文件扩展名，且必须安装 io 包（键入“pkg install -forge io”即可完成）。首列可包含每个观测值的日期。
- CSV 文件（扩展名为*.csv）：对于每个内生变量和外生变量，文件中必须包含一个同名列。首列可以包含每个观察值的日期。

first_obs={INTEGER|DATE}：文件中要使用的首个观察数据的编号或日期（参见 6.1.2 日期类）。

first_simulation_period={INTEGER|DATE}：文件中的观测数据或模拟（或预测）开始的日期（参见 *dates*）。这个选项可以避免计算模型中的最大滞后数。对应于首个模拟期的观测值不需要存在文件中，因为初始化所需的唯一日期在该日期之前。

last_obs={INTEGER|DATE}：文件中要使用的最后一个观测数据的编号或日期（参见 6.1.2 日期类）。

nobs=INTEGER：文件中要使用的观测值数量（从 first_obs 的首个观测值开始）。

series=DSERIES NAME：包含数据的 DSERIES 的名称（参见 6.2 dseries 类）。

示例 1

```
var c x;
varexo e;
parameters a b c d;

a=1.5;
b=-0,6;
c=0.5;
d=0.5;

model;
x=a*x(-1)+b*x(-2)+e;
log(c)=c*x+d*log(c(+1));
end;

steady_state_model;
x=0;
c=exp(c*x/(1-d));
end;

histval_file(datafile=mydata.csv);

stoch_simul(order=1,periods=100);
```

随机模拟的初始值取自文件 mydata.csv 的前两行。

示例 2

```
var c x;
varexo e;
parameters a b c d;

a=1.5;
b=-0,6;
c=0.5;
d=0.5;
```

```
model;
x=a*x(-1)+b*x(-2)+e;
log(c)=c*x+d*log(c(+1));
end;

histval_file(datafile=mydata.csv,first_obs=10);

stoch_simul(order=1,periods=100);
```

随机模拟的初始值取自文件 mydata.csv 的第 10 和 11 行。

示例 3

```
var c x;
varexo e;
parameters a b c d;

a=1.5;
b=-0,6;
c=0.5;
d=0.5;

model;
x=a*x(-1)+b*x(-2)+e;
log(c)=c*x+d*log(c(+1));
end;

histval_file(datafile=mydata.csv,first_obs=2010Q1);

stoch_simul(order=1,periods=100);
```

随机模拟的初始值取自文件 mydata.csv 的 2010Q1 和 2010Q2 观测值。

示例 4

```
var c x;
varexo e;
parameters a b c d;
```

```
a=1.5;
b=-0,6;
c=0.5;
d=0.5;

model;
x=a*x(-1)+b*x(-2)+e;
log(c)=c*x+d*log(c(+1));
end;

histval_file(datafile=mydata.csv,first_simulation_period=2010
Q1)

stoch_simul(order=1,periods=100);
```

随机模拟的初始值取自文件 mydata.csv 的 2009Q3 和 2009Q4 的观测值。

示例 5

```
var c x;
varexo e;
parameters a b c d;

a=1.5;
b=-0,6;
c=0.5;
d=0.5;

model;
x=a*x(-1)+b*x(-2)+e;
log(c)=c*x+d*log(c(+1));
end;

histval_file(datafile=mydata.csv,last_obs=4);

stoch_simul(order=1,periods=100);
```

随机模拟的初始值取自文件 mydata.csv 的前两行。

示例 6

```
var c x;
varexo e;
parameters a b c d;

a=1.5;
b=-0,6;
c=0.5;
d=0.5;

model;
x=a*x(-1)+b*x(-2)+e;
log(c)=c*x+d*log(c(+1));
end;

initval_file(datafile=mydata.csv,first_obs=10,nobs=4);

stoch_simul(order=1,periods=100);
```

随机模拟的初始值取自文件 mydata.csv 的第 10 和 11 行。

示例 7

```
var c x;
varexo e;
parameters a b c d;

a=1.5;
b=-0,6;
c=0.5;
d=0.5;

model;
x=a*x(-1)+b*x(-2)+e;
log(c)=c*x+d*log(c(+1));
end;
```

```
initval_file(datafile=mydata.csv,first_obs=10);

histval_file(datafile=myotherdata.csv);

perfect_foresight_setup(periods=200);
perfect_foresight_solver;
```

模拟的历史初始值取自文件 myotherdata.csv 的前两行，终端值和猜测值取自文件 mydata.csv，从第 12 个观测值开始。文件中至少要有 212 个观测值。

4.8 外生变量冲击

确定性背景下，模型从一个均衡转向另一个均衡，相当于分析永久冲击的后果，在 Dynare 中正确使用 initial 和 endval 即可实现。另一种典型的实验是研究系统回到原始均衡状态（如果模型是稳定的……）的暂时性冲击。暂时性冲击是模型的一个或多个外生变量暂时变化，由 shocks 声明。

随机框架下，外生变量在每期都取随机值。Dynare 的随机值遵循零均值的正态分布，但是用户可以设定冲击的波动性。冲击的方差－协方差矩阵的非零元素可以用 shocks 命令输入，或者用 Sigma_e 直接输入整个矩阵（该用法现已废弃）。

如果外生变量的方差设置为零，该变量将出现在策略和转移函数的结果中，但是不用于矩和脉冲响应函数的计算。设置方差为零是消除外生冲击的简单方法。

注意，默认情况下，如果在同一个*.mod 文件中存在多个 shocks 或 mshocks 模块，则把它们累积起来，考虑所有模块中声明的所有冲击；但是，只要有一个 shocks 或 mshocks 模块使用了 overwrite 选项，则替换所有前面的 shocks 和 mshocks 模块。

```
Block:shocks;
Block:shocks(overwrite);
```

参见前文的 overwrite 选项的含义。

在确定性模型中

确定性模拟的 shocks 模块声明外生变量的暂时性变化，永久性冲击使用 endval。

该模块应该包含以下三行的一组或多组命令：

```
var VARIABLE_NAME;
periods INTEGER[:INTEGER] [[,] INTEGER[:INTEGER]]...;
values DOUBLE | (EXPRESSION) [[,] DOUBLE | (EXPRESSION)]...;
```

同时也可以声明冲击持续且随时间变化的时期数。关键词 periods 可以接受多个日期或日期范围的列表，这些日期或日期范围必须匹配关键词 values 相同个数的冲击值。注意，关键词 periods 的一个范围只能匹配关键词 values 的一个值。如果 values 表示标量，则相同的值适用于整个日期范围。如果 values 表示向量，则必须具有与日期范

围相同数量的元素。

注意，冲击值不限于数值常数，也可以使用任意表达式，但必须将它们写在括号内。periods 的可行范围是从 0 到 perfect_foresight_setup 中指定的 periods 数量。

警告：注意，首个内生模拟期是第 1 期。因此，初始期 0 指定的冲击值可能与 initval 或 endval（取决于具体环境）指定的初始期数值相冲突（即可能覆盖或被覆盖）。用户应该在 perfect_foresight_setup 之后验证 oo_.exo_simul 的设置是否正确。

示例（标量值）

```
shocks;

var e;
periods 1;
values 0.5;
var u;
periods 4:5;
values 0;
var v;
periods 4:5 6 7:9;
values 1 1.1 0.9;
var w;
periods 1 2;
values (1+p) (exp(z));

end;
```

示例（向量值）

```
xx=[1.2;1.3;1];

shocks;
var e;
periods 1:3;
values (xx);
end;
```

在随机性模型中

对于随机模拟，shocks 模块声明了外生变量冲击的协方差矩阵的非零元素。可以在模块中使用以下类型的条目：

- 声明外生变量标准误：

```
var VARIABLE_NAME;stderr EXPRESSION;
```

- 声明外生变量方差：

```
var VARIABLE_NAME=EXPRESSION;
```

- 声明两个变量的协方差：

```
var VARIABLE_NAME, VARIABLE_NAME=EXPRESSION;
```

- 声明两个外生变量的相关系数。

```
corr VARIABLE_NAME, VARIABLE_NAME=EXPRESSION;
```

模型估计中还可以声明内生变量的方差和协方差，这些值解释为变量的测量误差的校准值，需要在 shocks 模块之前用 varobs 命令来声明。

示例

```
shocks;
var e=0.000081;
var u; stderr 0.009;
corr e, u=0.8;
var v, w=2;
end;
```

在随机最优策略背景下

在 ramsey_model 或 discretionary_policy 背景下计算条件福利时，福利是以计划者在第一阶段做出选择时继承的状态值为条件。第一阶段的信息集包括各自的外生冲击的实现。因此，它们的已知值可以用完全预见的语法来指定。需要注意：（1）除第一期以外的所有其他期设定的数值被忽略；（2）滞后冲击的数值（例如在消息冲击的情况下）用 histval 指定。

示例

```
shocks;
 var u; stderr 0.008;
 var u;
 periods 1;
 values 1;
end;
```

确定性冲击和随机冲击混合

可以混合确定性和随机性冲击来建立模型，代理人从模拟开始就知道未来的外生变化。stoch_simul 将把未来信息添加到状态空间中来计算理性预期解（stoch_simul 的输出中没有显示任何信息），并且 forecast 将基于初始条件和未来信息来计算模拟值。

示例

```
varexo_det tau;
varexo e;
...
shocks;
var e; stderr 0.01;
var tau;
periods 1:9;
values -0.15;
end;

stoch_simul(irf=0);

forecast;
```

Block:mshocks;

Block:mshocks(overwrite);

确定性冲击下，除了只是给出的数值将以乘法方式解释，这个模块的用途类似于 shocks。例如，如果在某个时期将 1.05 的值作为某些外生冲击的大小，则意味着比其稳态值（如 initval 或 endval 模块给出）高出 5%。该模块的语法与确定性环境下的 shocks 是相同的，在两种情况下才有意义：

- 在确定环境中，具有非零稳态的外生变量；
- 在随机环境中，具有非零稳态的确定性外生变量。

参见前文的 overwrite 选项的含义。

Block:heteroskedastic_shocks;

Block:heteroskedastic_shocks(overwrite);

在估计方面，它实现了异方差滤波器，冲击标准误差在每个时期都可能意外地改变。冲击的标准差可以直接提供，也可以在每个观察期通过比例因子设置/修改。如果 std0 是 shock1 的通常标准误差，那么：

- 在 t 期使用比例因子意味着 std(shock1|t)=std0(shock1)*scale(t);
- 在 t 期使用提供的数值意味着 std(shock1|t)=value(t)。

该模块的语法与完美预期背景下的 shock 相似，应该包含一个或多个以下三行组的出现（用于设置值）：

```
var VARIABLE_NAME;
periods INTEGER[:INTEGER] [[,] INTEGER[:INTEGER]]...;
values DOUBLE | (EXPRESSION)  [[,] DOUBLE | (EXPRESSIO
```

```
N) ]...;
```

或（用于设置比例因子）：

```
var VARIABLE_NAME;
periods INTEGER[:INTEGER] [[,] INTEGER[:INTEGER]]...;
scales DOUBLE | (EXPRESSION)  [[,] DOUBLE | (EXPRESSIO
N) ]...;
```

注意：scales 和 values 不能同时设置为同一时期的同一冲击，但可以为某些时期设置 values，为同一冲击的其他时期设置 scales。一个给定的冲击只能有一个 scales 和 values 指令，所以所有受影响的时期必须在一个语句中设置。

示例

```
heteroskedastic_shocks;

var e1;
periods 86:87, 89:97;
scales 0.5, 0;

var e1;
periods 88;
values 0.1;

var e2;
periods 86:87 88:97;
values 0.04 0.01;

end;
```

Special variable:Sigma_e

这个特殊变量直接声明了随机冲击的协方差矩阵为上（或下）三角形矩阵。Dynare 建立相应的对称矩阵。除了最后一个，三角矩阵的每一行必须用分号终止。给定的元素允许使用任意的表达式（而不是简单的常量），但是需要将表达式括在括号中。矩阵中协方差的顺序与 varexo 模块中声明的顺序相同。

示例

```
varexou,e;

Sigma_e=[0.81(phi*0.9*0.009);
```

```
0.000081];
```

上例将 u 的方差设为 0.81，e 的方差为 0.000081，e 与 u 之间的相关系数为 phi。

警告：这种特殊变量的使用已被弃用，强烈建议不要使用，应该用 shocks 模块代替。

MATLAB/Octave command:get_shock_stderr_by_name('EXOGENOUS_NAME');

给定一个外生变量的名称，返回它的标准差，如前一个 shocks 模块所设定的。

MATLAB/Octave command:set_shock_stderr_value('EXOGENOUS_NAME',MATLAB_EXPRESSION);

设置一个外生变量的标准偏差。这与通过 shocks 模块设置标准误差的作用基本相同，只是它接受任意的 MATLAB/Octave 表达式，并且可以从 MATLAB/Octave 脚本中工作。

4.9 其他一般性声明

Command:dsample INTEGER[INTEGER];

减少后续输出命令中所考虑的期数。

Command:periods INTEGER;

现已废弃（但仍然适用于旧的模型文件），没有模拟时不是必选项，取而代之的是在 perfect_foresight_setup、simul 和 stoch_simul 中使用 periods 选项。

此命令设置模拟从 1 到某个整数的期数。完美预期模拟假设所有未来事件都在第 1 期就已经完全知晓。

示例

```
periods 100;
```

4.10 稳态

计算模型的稳态（即静态均衡）有两种方法：一是使用非线性牛顿型算法。该方法适用于大多数模型，并且使用起来相对简单；二是根据您对模型的熟悉程度，通过提供计算方法，使用 steady_state_model 模块或编写 matlab 路径，引导 Dynare 计算稳态。

4.10.1 用 Dynare 非线性算法解稳态

Command:steady;

Command:steady(OPTIONS...);

使用非线性牛顿型算法计算模型稳态并显示结果。使用稳态文件时，steady 展示稳态，并检验是否是静态模型的解。更准确地说，前文论述的 initval 和 endval 模块已声明外生变量值的情况下，它计算出内生变量的均衡值。

steady 使用迭代过程，把前文 initval 或 endval 模块声明的值作为内生变量的初始猜测值。

对于复杂的模型，为内生变量找到良好的数值初始值是计算模型均衡值最棘手的部分。通常最好从一个较小的模型开始，然后逐个添加新变量。

选项

maxit=INTEGER：确定非线性算法中使用的最大迭代次数。maxit 的默认值为 50。

tolf=DOUBLE：基于函数值的终止迭代的收敛标准。当残差小于 tolf 时，迭代将停止。默认值：eps^(1/3)

solve_algo=INTEGER：确定要使用的非线性算法。该选项的可能值有如下几种：

0：使用 fsolve（MATLAB 只有安装了优化工具箱才可用，Octave 中总是可用）；

1：使用 Dynare 自带的非线性方程求解程序（线性搜索的牛顿型算法）；

2：将模型分割成递归模块，并使用与算法 1 相同的求解程序依次解决每个模块；

3：使用 Chris Sims 的求解程序；

4：将模型分割成递归模块，并使用具有自动缩放功能的信赖域（trust region）算法程序依次解决每个模块；

5：具有稀疏高斯消元（SPE）的牛顿算法（需要 bytecode 选项，参见 4.5 模型声明）；

6：每次迭代时使用具有与稀疏 LU 求解法的牛顿算法（需要 bytecode 和/或 block 选项，参见 4.5 模型声明）；

7：在每次迭代时，使用具有广义最小残差（GMRES）求解的牛顿算法（需要 bytecode 和/或 block 选项，参见 4.5 模型声明）；

8：在每次迭代中，使用具有稳定双共轭梯度（BICGSTAB）求解的牛顿算法（需要 bytecode 和/或 block 选项，参见 4.5 模型声明）；

9：整个模型上的信赖域算法；

10：Levenberg-Marquardt 混合互补问题（LMMCP）求解（Kanzow 和 Petra，2004）；

11：Ferris 和 Munson（1999）的 PATH 混合互补问题算法，互补条件用 mcp 方程标签来指定，参见 *lmmcp*。Dynare 只提供使用该算法的接口。由于许可限制，必须从 http://pages.cs.wisc.edu/~ferris/path.html 下载该算法的最新版本，并放置在 MATLAB 的搜索路径中；

12：为算法 2 的特殊版本，适用于所有方程左侧都有一个内生变量且每个方程确定不同内生变量的模型。算法要求方程左侧的表达形式必须是内生变量的自然对数，或者是内生变量的一阶差分（使用 diff 运算符），或者是内生变量对数的一阶差分。通过计算右侧的表达式可以求解单变量模块；

14：为算法 4 的特殊版本，适用于所有方程左侧都有一个内生变量且每个方程确定不同内生变量的模型。算法要求方程左侧的表达形式必须是内生变量的自然对数，或者是内生变量的一阶差分（使用 diff 运算符），或者是内生变量的对数的一阶差分。通过计算右侧的表达式可以求解单变量模块进行。

默认值：4。

homotopy_mode=INTEGER：使用同伦算法（或 divide-and-conquer）技术求解稳态。

使用此选项必须声明一个 hometopy_setup 模块。此选项可以采用以下三种可能的值：

1：在这种模式下，所有的参数都是同时变化，并且参数边界之间的距离被划分为多个间隔区间，而且区间数与步数一致（步数由 homotopy_steps 选项定义），问题也需要与步数一样多的次数求解。

2：与模式 1 相同，只是每次只改变一个参数，求解的次数就是步数乘以参数个数。

3：Dynare 首先尝试最极端的值。如果不能计算稳态，则初始值和合意值之间的就划分为两段。只要找不到稳态，前一个区间就再次划分为两段。成功找到稳态，前一个间隔乘以 2。最后一种情况下，homomotopy_steps 包含放弃之前尝试的最大计算次数。

homotopy_steps=INTEGER：定义执行同伦算法的步骤数。详情参见 *homotopy_mode* 选项。

homotopy_force_continue=INTEGER：控制同伦算法失败时发生的情况。

0：显示 steady 失败的错误信息；

1：Steady 保持成功并继续的最后一个同伦算法步骤的次数。特别注意：参数和/或外生变量不是用户期望的值。

默认值：0。

nocheck：当稳态文件或 steady_state_model 模块明确提供稳态值时，不要检查稳态。对于具有单位根的模型很有用，此时稳态不唯一或不存在。

markowitz=DOUBLE：马科维茨准则的值，用于选择枢轴量。仅在 solve_algo=5 时使用。默认值：0.5。

示例

参见 4.7 初始条件和终止条件。

计算之后，下列变量可以得到稳态：

MATLAB/Octave variable:oo_.steady_state

包含计算的稳态。内生变量按照 var 命令中使用的声明顺序排列（这也是 M_.endo_names 使用的顺序）。

MATLAB/Octave command:get_mean('ENDOGENOUS_NAME'[,'ENDOGENOUS_NAME']...);

返回给定内生变量的稳态，存储在 oo_.steady_state 中。注意，如果尚未使用 steady 计算稳态，它将首先尝试计算。

Block:homotopy_setup;

使用同伦算法时，该模块声明初始值和终值，与稳态命令的 homotopy_mode 选项一起使用。

同伦算法的思想（也被一些作者称为 divide-and-conquer）是将寻找稳态的问题细分为更小的问题。已知如何计算给定参数集下的稳态，通过逐步地从一个参数集移到另一个参

数集，进而帮助找到另一组参数集下的稳态。

homotopy_setup 模块的目的是声明在同伦算法期间将更改的参数或外生变量的终值（也可能是初值），应该包含以下命令行：

```
VARIABLE_NAME,EXPRESSION,EXPRESSION;
```

该句法声明了给定的参数/外生变量的初值和终值。

还有一个备择句法：

```
VARIABLE_NAME,EXPRESSION;
```

这里仅为给定的参数/外生变量指定了终值，初始值取自前面的 initval 模块。

成功的同伦算法一个必要条件是，在无需任何帮助下，Dynare 必须能够求解初始参数/外生变量下的稳态（使用在 initval 模块中给出的猜测值）。如果同伦算法失败，可能的解决方案是增加步数（在 steady 的 homotopy_steps 选项下设定）。

示例

如下示例中，Dynare 首先计算初始值下的稳态（gam=0.5 和 x=1），然后将问题细分为 50 个较小的问题以找到终值下的稳态（gam=2 和 x=2）。

```
var c k;
varexo x;

parameters alph gam delt bet aa;
alph=0.5;
delt=0.02;
aa=0.5;
bet=0.05;

model;
c+k-aa*x*k(-1)^alph-(1-delt)*k(-1);
c^(-gam)-(1+bet)^(-1)*(aa*alph*x(+1)*k^(alph-1)+1-delt)*c(+1)
^(-gam);
end;

initval;
x=1;
k=((delt+bet)/(aa*x*alph))^(1/(alph-1));
c=aa*x*k^alph-delt*k;
end;
```

```
homotopy_setup;
gam,0.5,2;
x,2;
end;

steady(homotopy_mode=1,homotopy_steps=50);
```

4.10.2 向 Dynare 提供稳态

如果您知道如何计算模型的稳态，可以提供一个 MATLAB/Octave 函数计算，而不是使用 steady 命令。同样有两种选择：

- 最简单的方法是编写一个 steady_state_model 块，下面将对其进行更详细的描述。另参见示例目录中的 fs2000.mod 以获取示例。Dynare 生成的稳态文件将被称为+FILENAME/steadystate.m；
- 手工编写相应的 MATLAB 函数。如果 MOD 文件名称为 FILENAME.mod，则稳态文件必须命名为 FILENAME_steadystate.m。参见示例目录中 NK_baseline_steadystate.m。这个选项提供了更多的灵活性（可以使用循环和条件结构），但代价是更重的程序编程负担和更低的计算效率。

注意，这两个文件都允许调用函数时更新参数。例如，通过设置劳动负效用参数等于相应的值，从而将稳态劳动供给校准为 0.2（参见 example 目录中的 NK_baseline_steadystate.m）。它们也可以用于估计，其中一些参数可能是待估参数的函数，并且需要针对每个参数抽样更新。例如，人们可能想把资本利用成本参数设置为贴现率的函数，以确保稳态下的利用率为 1。设两个参数互相独立或一个参数为另一个参数的函数而不更新的话，将导致错误的结果。但是，这也意味着需要格外小心，不要用新的值覆盖参数，因为会导致错误的结果。

Block:steady_state_model;

模型的解析解已知时，可以更有效和可靠帮助 Dynare 找到稳态，特别是在估计过程中必须对参数空间中的每个点重新计算稳态。

模块的每一行命令都包含一个变量（内生变量、临时变量或参数），该变量被赋值为一个表达式（可以包含稳态下的参数、外生变量，也可以包含上面已经声明的任何内生变量或临时变量）。因此，每一行命令格式如下：

```
VARIABLE_NAME=EXPRESSION;
```

注意，如果右侧的主函数是返回多个参数的 MATLAB/Octave 函数，则也可以同时分配多个变量：

```
[VARIABLE_NAME,VARIABLE_NAME...]=EXPRESSION;
```

Dynare 将使用此模块提供的信息自动生成一个稳态文件（格式为+FILENAME_steadystate2.m）。

确定性模型的稳态文件

steady_state_model 模块在确定性模型中也可以使用。必要时，initval 和 endval 模块设置外生变量的值。每个 initval 或 endval 模块后面都必须跟着 steady 以执行 steady_state_model 创建的函数，并分别设置初始稳态和最终稳态。

示例

```
var m P c e W R k d n l gy_obs gp_obs y dA;
varexo e_a e_m;

parameters alp bet gam mst rho psi del;

...
//parameter calibration, (dynamic) model declaration, shock c
alibration...
...

steady_state_model;
  dA=exp(gam);
  gst=1/dA;//Atemporaryvariable
  m=mst;

  //Three other temporary variables
  khst=((1-gst*bet*(1-del))/(alp*gst^alp*bet))^(1/(alp-1));
  xist=(((khst*gst)^alp-(1-gst*(1-del))*khst)/mst)^(-1);
  nust=psi*mst^2/((1-alp)*(1-psi)*bet*gst^alp*khst^alp);

  n=xist/(nust+xist);
  P=xist+nust;
  k=khst*n;

  l=psi*mst*n/((1-psi)*(1-n));
  c=mst/P;
  d=l-mst+1;
```

```
  y=k^alp*n^(1-alp)*gst^alp;
  R=mst/bet;

  //You can use MATLAB functions which return several argumen
ts
  [W,e]=my_function(l,n);

  gp_obs=m/dA;
  gy_obs=dA;
end;

steady;
```

4.10.3 稳态计算替换一些方程

没有使用稳态文件时，Dynare 通过求解静态模型来计算稳态，即从*.mod 文件中去掉超前和滞后期的模型。

在某些特定的情况下，人们可能希望对这种静态模型的创建方式有更多的控制。因此，Dynare 提供了明确给出静态模型中方程形式的可能性。

更准确地说，如果一个方程是由[static]标签预先设定的，那么它将出现在用于稳态计算的静态模型中，但是这个方程不会用于其他计算。对于以这种方式标记的每个方程，您必须用[dynamic]标记另一个方程：该方程将不用于稳态计算，而是用于其他计算。

此功能在具有单位根的模型上很有用，其中有无限个稳态。方程（标记为[dynamic]）将给出非平稳变量的运动定律（如随机游走）。为了确定一个特定的稳态，标记为[static]的方程会影响非平稳变量的恒定值。[static]标签可能有用的另一种情况是当人们只有稳态的部分封闭形式解决方案时。

示例

这是一个具有两个内生变量的简单例子。第二个方程在静态模型中采用不同的形式。

```
var c k;
varexo x;
...
model;
c+k-aa*x*k(-1)^alph-(1-delt)*k(-1);
[dynamic]c^(-gam)-(1+bet)^(-1)*(aa*alph*x(+1)*k^(alph-1)+1-de
lt)*c(+1)^(-gam);
[static]k=((delt+bet)/(x*aa*alph))^(1/(alph-1));
```

```
end;
```

4.11 获取模型信息

Command:check;

Command:check(OPTIONS…);

计算最后一个 initval、endval 或 stable 语句附近指定的数值线性化的模型特征值。一般来说，特征值仅在模型在稳态附近线性化后才有意义的。它是一种稳态邻域内进行局部分析的方法。

稳态附近存在唯一稳定均衡的必要条件是系统中大于 1 的模的特征值与前向变量的数量相同。附加秩条件要求对应的前向变量（跳跃变量）和爆炸特征值对应的右 Schur 向量的方阵子矩阵满秩。

注意，当特征值非常接近 *qz_criterium* 时，结果可能异于 sum(abs(oo_.dr.eigval))。

选项

solve_algo=INTEGER：对于可能的选项及其含义，参见 *solve-algol*。

qz_zero_threshold=DOUBLE：用于测试广义 Schur 分解中广义特征值是否为 0/0 的形式（此时模型没有唯一解）。默认值：1e-6。

输出

在全局变量 oo_.dr.eigval 中 check 得到的特征值。

MATLAB/Octave variable:oo_.dr.eigval

包含由 check 命令计算的模型的特征值。

Command:model_diagnostics;

对模型执行各种完整性检查，如果发现问题（丢失当前时期变量，无效稳态，静态模型的奇异雅可比行列式）就显示相关信息。

Command:model_info;

Command:model_info(OPTIONS...);

此命令提供如下信息：

当不在 model 模块的 block 选项的环境中使用此命令时，它会提供预定状态变量、前向变量和纯静态变量的列表。

在 model 模块 block 选项的环境中使用此命令时会显示：

- 模型的规范化：内生变量归属于模型的每个方程；
- 模型的模块结构：对于每个模块，model_info 都会表示出它的类型，以及属于该模块的方程数和内生变量数。

根据使用方法，有五种不同类型的选项：

- EVALUATE FORWARD：模块包含的方程具有如下特征：归属于方程的内生变量

出现在左侧和当期；没有前向内生变量出现。形式为$y_{j,t} = f_j(y_t, y_{t-1}, \dots, y_{t-k})$；

- EVALUATE BACKWARD：模块包含的方程具有如下特征：归属于方程的内生变量出现在左侧和当期；没有后向内生变量出现。形式为$y_{j,t} = f_j(y_t, y_{t+1}, \dots, y_{t+k})$；
- SOLVE BACKWARD x：模块包含的方程具有如下特征：归属于方程的内生变量没有出现在左侧和当期；没有前向内生变量出现。形式为$g_j(y_{j,t}, y_t, y_{t-1}, \dots, y_{t-k}) = 0$。如果模块只有一个方程，那么 x 等于 SIMPLE，多个方程时 x 等于 COMPLETE；
- SOLVE FORWARD x：模块包含的方程具有如下特征：归属于方程的内生变量没有出现在左侧和当期；没有后向内生变量出现。形式为$g_j(y_{j,t}, y_t, y_{t+1}, \dots, y_{t+k}) = 0$。模块只有一个方程，x 等于 SIMPLE，多个方程时 x 等于 COMPLETE；
- SOLVE TWO BOUNDARIES x：模块包含的方程取决于前向变量和后向变量。形式为$g_j(y_{j,t}, y_t, y_{t-1}, \dots, y_{t-k}, y_t, y_{t+1}, \dots, y_{t+k}) = 0$。模块只有一个方程，x 等于 SIMPLE。如果模块中出现了几个方程，x 等于 COMPLETE。

选项

static：打印静态模型的模块分解。没有 static 选项，model_info 显示动态模型的模块分解。

incidence：显示模块分解模型的总关联矩阵和重排关联矩阵。

Command:print_bytecode_dynamic_model;

打印存储在 bytecode 二进制格式文件中的动态模型方程及雅可比行列式。只能与 model 模块的 bytecode 选项一起使用。

Command:print_bytecode_static_model;

展示存储在 bytecode 二进制格式文件中的静态模型方程和雅可比行列式。只能与 model 模块的 bytecode 选项一起使用。

4.12 确定性模拟

当模型是确定性的，Dynare 适用于完美预期假设的模型。通常情况下，当模型中的代理人获悉同期或者未来消息之前的时期“1”，系统应该处于均衡状态。模拟的目的是描述预期和冲击的响应，直到系统回到旧的或新的均衡状态。大多数模型的均衡回归只是一种渐近现象，只有模拟时间足够长才能近似得到。

非常适合分析 Dynare 的另一项案例是研究在永久性冲击之后向新均衡转移的路径。对于确定性模拟，数值问题就是求解n个内生变量T期的非线性方程组。Dynare 提供了几种解决这个问题的算法，可以通过 stack_solve_algo 选项设定。默认情况下（stack_solve_algo=0），Dynare 使用牛顿型方法来求解联立方程系统。因为雅可比行列式的阶数为 n×T，对于具有多个变量的长时间模拟而言工作量很大。Dynare 利用 MATLAB/Octave 的稀疏矩阵能力；第二种较慢但可能较少消耗内存的替代算法（stack_solve_algo=6）

是基于 Laffargue（1990）和 Boucekkine（1995）首先提出的牛顿型算法，该算法使用松弛技术，因此避免了存储完整的雅可比行列式，细节参见 Juillard（1996）中找到；第三种类型的算法利用模块分解技术（divide-and-conquer methods）分析模型结构，原理是识别模型结构中的递归和联立模块，并使用此信息协助求解。这些算法可以显著提速求解大型模型。

警告：完美预期的情况下计算时慎用辅助变量。相同的模型可能适用于随机模拟，但不适用于完美预期模拟。当方程突然只包含日期为 t+1（或 t-1）的变量时就会出现问题。在这种情况下，末期（或首期）相应的所有变量的导数为 0，呈现堆叠的雅可比奇异值。

示例

考虑以下包含对数效用的欧拉方程的情况：

```
Lambda=beta*C(-1)/C;
Lambda(+1)*R(+1)=1;
```

显然，第二个方程中所有内生变量在时间 t 的导数是零，导致 perfect_foresight_solver 基本失效，因为使用拉格朗日乘子 Lambda 作为辅助变量。相反，采用相同的

```
beta*C/C(+1)*R(+1)=1;
```

则会成功。

Command:perfect_foresight_setup;

Command:perfect_foresight_setup(OPTIONS...);

通过提取 initval，endval 和 shocks 模块的信息，转换为外生变量和内生变量的模拟路径，从而准备完美预期模拟。运行 perfect_foresight_solver 模拟之前，必须调用此命令。

选项

periods=INTEGER：模拟期数。

datafile=FILENAME：为所有内生变量和外生变量指定路径，严格等价于 *initval_file*。

输出

外生变量的路径存储在 oo_.exo_simul 中。

内生变量的初始条件和终止条件、初始猜测值路径存储在 oo_.endo_simul。

Command:perfect_foresight_solver;

Command:perfect_foresight_solver(OPTIONS...);

计算模型的完美预期（或确定性）模拟。

注意，此命令之前必须先调用 perfect_foresight_setup 才能设置模拟环境。

选项

maxit=INTEGER：确定非线性解算法中最大迭代次数。默认值：50。

tolf=DOUBLE：基于函数值的终值收敛准则。当证明超过 tolf 的方式改善函数值

不成立时，停止迭代。默认值：1e-5。

tolx=DOUBLE：基于函数参数变化的终值收敛标准。当求解算法尝试采用小于 tolx 的步骤时，停止迭代。默认值：1e-5。

noprint：不要打印任何结果，对循环很有用。

print：打印结果（和 noprint 相对）

stack_solve_algo=INTEGER

求解的代数方法，可能的值有：

0：使用稀疏矩阵同时求解每期的所有方程的牛顿型算法（默认）；

1：每次迭代时使用带有稀疏 LU 求解的牛顿型算法（需要 bytecode 和/或 block 选项，参见 4.5 模型声明）；

2：每次迭代时使用广义最小残差（GMRES）求解的牛顿型算法（需要 bytecode 和/或 block 选项，参见 4.5 模型声明）；

3：每次迭代时使用稳定双共轭梯度的（BICGSTAB）求解牛顿型算法（需要 bytecode 和/或 block 选项，参见 4.5 模型声明）；

4：每次迭代时使用最佳路径长度的牛顿型算法（需要 bytecode 和/或 block 选项，参见 4.5 模型声明）；

5：每次迭代时使用稀疏高斯消元（SPE）求解的牛顿型算法（需要 bytecode 选项，参见 4.5 模型声明）；

6：使用 Juillard（1996）提出的历史算法：比 stack_solve_algo=0 慢，但在大型模型上可能消耗更少的内存（不适用于 bytecode 和/或 block 选项）；

7：允许用户通过选项 solve_algo 来使用可行的算法解完美预期模型（参见 *solve_algo* 获取可能值的列表，注意，需要 bytecode 和/或 block 选项的 5、6、7 和 8 是不允许的）。例如，以下命令：

```
perfect_foresight_setup(periods=400);
perfect_foresight_solver(stack_solve_algo=7,solve_algo=9)
```

使用信赖域算法触发解的计算。

robust_lin_solve：对默认的 stack_solve_algo=0 而言，触发使用稳健线性求解算法。

solve_algo：参见 *solve_algo*。允许选择与 stack_solve_algo=7 并用求解算法。

no_homotopy：默认情况下，如果算法无法求解，完美预期求解就使用同伦算法。具体地说，它通过减小冲击的大小并逐渐增加冲击，将问题细分，直至收敛。此选项告诉 Dynare 禁用该行为。注意，纯前向或后向模型不能实施同伦算法。

markowitz=DOUBLE：用于选择枢轴量的 Markowitz 标准值。仅用于 stack_solve_algo=5。默认值：0.5。

minimal_solving_periods=INTEGER：对其余周期使用一组常量运算之前，指定解出模型的最小期数。仅用于 stack_solve_algo=5。默认值：1。

lmmcp：使用 Levenberg-Marquardt 混合互补问题（LMMCP）算法（Kanzow 和 Petra，2004）求解完美预期模型，该算法允许考虑内生变量的不等式约束（例如名义利率的 ZLB 或不可逆投资的模型）。此选项等价于 stack_solve_algo=7 和 solve_algo=10。使用 LMMCP 算法需要特殊的模型设置，因为目标是摆脱任何最小/最大算子和互补松弛条件，这些算子和互补松弛条件可能会在雅可比行列式中引入奇异点。这是通过将 mcp 关键字附加到受影响的方程标记（参见 4.5 模型声明）完成的。此标记表明，除非标记内的表达式具有约束力，否则附加标记的等式必须保留。例如，名义利率的 ZLB 将在 model 模块中指定如下：

```
model;
    ...
    [mcp='r>-1.94478']
    r=rho*r(-1)+(1-rho)*(gpi*Infl+gy*YGap)+e;
    ...
end;
```

其中 1.94478 是名义利率的稳态水平，r 是偏离稳态的名义利率。该结构意味着泰勒规则是有效的，除非隐含利率 r<=-1.94478，在这种情况下，r 固定在-1.94478（因此等于互补松弛条件）。通过约束来自该等式的 r 值，mcp 的标记还避免 r 在模型的其余部分使用 max（r，-1.94478）。特别关键的是，因为 mcp 的标记有效地取代了互补松弛条件，所以不能简单地附加到任何方程。相反，它必须附加到正确的受影响的方程，否则求解算法计算一个不同于原问题的新问题。此外，由于求解的问题是非线性的，因此动态方程的残差符号很重要。在上述例子中，对于名义利率规则，如果 LHS 和 RHS 换位，残差（LHS 和 RHS 之间的差值）变号化，求解算法可能无法识别求解路径。更一般地，使用相同方程的数学等价表示时，很难保证非线性求解算法的收敛性。

注意，当前应用的预处理程序不会解析 mcp 等式标记的内容。因此，不等式必须尽可能简单：一个内生变量，后面跟着关系运算符，然后是数字（不是变量，参数或表达式）。

endogenous_terminal_period：求解完美预期模型时，牛顿迭代中的期数不是恒定的。通过去除已经识别出解路径（和相关方程）的部分（直到公差参数），减小了非线性方程组的大小。这种策略可以解释为打靶法和松弛法混合。注意，该方法的舍入误差更重要（用户应检查最大绝对误差的报告值）。仅用于选项 stack_solve_algo==0。

linear_approximation：解决完美预期模型的线性化版本。模型必须平稳，仅用于选项 stack_solve_algo==0 和 stack_solve_algo==7。

输出

模拟的内生变量可在全局矩阵 oo_.endo_simul 获得。

Command:simul;

Command:simul(OPTIONS...);

触发模型的确定性模拟计算的缩减命令，完全等价于调用 perfect_foresight_setup，然后调用 perfect_foresight_solver。

选项

接受 perfect_foresight_setup 和 perfect_foresight_solver 的所有选项。

MATLAB/Octave variable:oo_.endo_simul

存储确定性模拟结果（perfect_foresight_solver 或 simul 计算）或随机模拟的结果（stoch_simul 使用 periods 选项或 extended_path 计算）。变量按声明的顺序逐行排列（如 M_.endo_names 所示）。注意，此变量还包含初始条件和终端条件，因此相应列数多于 periods 选项的值。

MATLAB/Octave variable:oo_.exo_simul

存储模拟期的外生变量路径（perfect_foresight_solver、simul、stoch_simul 或 extended_path 计算）。变量按声明顺序逐列排列（如 M_.exo_names 所示），期数按行排列。注意，列和行惯例与 oo_.endo_simul 的相反！

4.13 随机解和模拟

在随机环境下，Dynare 计算与一个冲击的随机过程对应的一个或多个模拟值。求解随机模型的主要算法依赖于期望函数的泰勒近似（最多三阶）（参见 Judd，1996；Collard 和 Juillard，2001a；Collard 和 Juillard，2001b 以及 Schmitt-Grohé 和 Uríbe，2004）。Villemot（2011）给出了 Dynare 实施一阶近似解的详细信息，使用 stoch_simul 命令求解。

作为替代方案，可以使用 Fair 和 Taylor（1983）提出的扩展路径（extended path）方法计算随机模型的模拟值，此方法特别适用于较强的非线性或约束条件场景。使用 extended_path 命令即可求解。

4.13.1 计算随机解

Command:stoch_simul[VARIABLE_NAME...];

Command:stoch_simul(OPTIONS...)[VARIABLE_NAME...];

使用扰动法求解随机（即理性预期）模型。更确切地说，stoch_simul 计算模型在确定性稳态附近的泰勒近似值，并求解近似模型的决策和转换函数。使用它可以计算脉冲响应函数和各种描述性统计（矩、方差分解、相关系数和自相关系数）。对于相关冲击，方差分解和 VAR 文献的一样，通过对外生变量的协方差矩阵的 Cholesky 分解来计算。当冲击相关时，方差分解取决于 varexo 命令中变量的顺序。

泰勒近似值是在稳态附近计算（参见 4.10 稳态）。IRFs 的计算方法是在第 1 期发生的冲击之后，变量的轨迹与其稳态值之间的差异。更多计算 IRFs 的细节参见 Dynare 百科。

具有严格正方差的变量才显示方差分解、相关系数、自相关系数。仅针对响应大于10^{-10}的变量才绘制脉冲响应函数图。

方差分解计算的是每个冲击的相对贡献，通常等于总方差。但如果模型产生非常大的方差，可能由于数值误差导致两者相去甚远。如果每个冲击的贡献总和与总方差之间的最大相对差异大于 0.01%，Dynare 会发出警告。

冲击的协方差矩阵在 shocks 模块声明（参见 4.8 外生变量的冲击）。

当声明 VARIABLE_NAME 列表时，仅显示这些变量的结果。

一阶近似的 stoch_simul 命令会受益于模型的模块分解（参见 *block*）。

选项

ar=INTEGER：计算和展示的自相关系数的阶数。默认值：5。

drop=INTEGER：在计算主要统计量之前，模拟开始时放弃的模拟点数量（burnin）。注意，此选项不会影响存储在 oo_.endo_simul 和工作区中的模拟数据序列。这里没有任何时间段被删除。默认值：100。

hp_filter=DOUBLE：计算矩之前使用参数λ=DOUBLE 的 HP 滤波。如果计算理论矩，按照 Uhlig（2001）概述的方法对模型解的频谱进行滤波处理。默认值：无滤波。

one_sided_hp_filter=DOUBLE：计算矩之前使用 Stock 和 Watson（1999）描述的λ=DOUBLE 的单边 HP 滤波。此选项仅适用于模拟矩。默认值：无滤波。

bandpass_filter：计算矩之前使用带有默认通带的带通滤波。如果要求理论矩，则使用理想带通滤波对模型解的频谱进行滤波处理。如果要求经验矩，使用 Baxter 和 King（1999）滤波。默认值：无滤波。

bandpass_filter=[HIGHEST_PERIODICITYLOWEST_PERIODICITY]：在计算矩之前，使用带通滤波。通带周期设置为 LOWEST_PERIODICITY，例如，如果模型频率是季度，则为 6 到 32 个季度。默认值：[6,32]。

filtered_theoretical_moments_grid=INTEGER：计算过滤理论矩时（使用选项 hp_filter 或 bandpass_filter），此选项控制离散快速傅立叶逆变换的网格中的点数。对于高度自相关的过程，可能需要增加数值。默认值：512。

irf=INTEGER：计算 IRFs 的周期数。设置 irf=0，禁止绘制 IRFs 图。默认值：40。

irf_shocks=(VARIABLE_NAME[[,]VARIABLE_NAME...])：计算 IRFs 的外生变量。默认值：全部。

relative_irf：执行标准化 IRFs 计算。一阶近似情形下，一单位冲击标准差的正常冲击向量除以当前冲击的标准差，再乘以 100。因此，脉冲响应对应 1 单位冲击（与常规的一个单位标准差相反）乘以 100。对于对数线性化模型——变量表示百分率——IRFs 就表示 100 个冲击的响应百分率。例如，产出对一单位 TFP 冲击的响应为 400，表示经济受到 100 个百分点的 TFP 冲击后，产出增加了 400 个百分点（TFP 增加了 100 个点）。给定

order=1 时的线性方程，可以直接将存储在 oo_.irfs 中的 IRF 重新缩放到任何所需的大小，更高阶近似的寒含义有所不同。然后，relative_irf 选项触发 IRFs 的生成，作为对 0.01 单位冲击的响应（对应于以百分比测量的冲击为 1 个百分点），也不乘以 100。也就是说，一单位冲击标准差的正常冲击向量除以当前冲击的标准差，再除以 100。例如，对数产出对单位 TFP 冲击的响应为 0.04（以稳态产出水平的百分比测量），表示在 1 个百分点的 TFP 冲击后，产出增加 4 个百分点（TFP 的冲击增加了 0.01）。

irf_plot_threshold=DOUBLE：绘制 IRFs 的阈值大小。某一变量的最大绝对离差水平小于这一阈值，IRFs 就不显示。默认值：1e-10。

nocorr：不显示相关系数矩阵。默认值：显示。

nodecomposition：不计算并且不显示无条件方差分解。

nofunctions：不显示近似解的系数。默认值：显示。

nomoments：不显示内生变量的矩。默认值：显示。

nograph：不要创建图形（这意味着不会显示，也不会保存到磁盘中）。如果未使用此选项，则图形将保存到磁盘（格式声明为 graph_format 选项，但 graph_format=none 除外），并显示到屏幕（除非使用 nodisplay 选项）。

graph：重启先前使用 nograph 关闭的生成图形功能。

nodisplay：不显示图形，但保存到磁盘（除非使用了 nograph）。

graph_format=FORMAT

graph_format=(FORMAT,FORMAT...)：指定保存到磁盘的图形的文件格式。可能的值是 eps（默认值）、pdf、fig 和 none（Octave 不承认 fig）。如果文件格式设置为 none，则会显示图形，但不会保存到磁盘。

noprint：参见 *noprint*。

print：参见 *print*。

order=INTEGER：泰勒近似阶数。注意，对于三阶及以上近似，隐含了 k_order_solver 选项，只有经验矩可用（必须为 periods 选项提供一个值）。默认值：2（除了 estimation 命令之后，因为此时的默认值是用于估计的数值）。

k_order_solver：使用*k*阶求解（用 C++实现）代替默认的 Dynare 泰勒近似阶数。此选项与 bytecode 尚不兼容（参见 4.5 模型声明）。默认值：1 和 2 无效，3 及以上有效。

periods=INTEGER：如果不是零，则计算经验矩而不是理论矩。选项的值指定模拟的期数。可能通过 steady 命令重新计算的 initval 模块的值，将被用作模拟的起始点。模拟的内生变量和外生变量分别在每个变量的向量和全局矩阵 oo_.endo_simul（以上内生变量，参见 *oo_.endo_simul*）、oo_.exo_simul（以上外生变量，参见 *oo_.exo_simul*）中提供给用户。默认值：0。

qz_criterium=DOUBLE：重新排序用于求解一阶问题的广义 Schur 分解，分割不稳

定特征值和稳定特征值。默认值：1.000001（使用 lik_init 选项等于 1 估计时除外，此时的默认值为 0.999999，参见 4.15 似然估计方法）。

qz_zero_threshold=DOUBLE：参见 *qz_zero_threshold*。

replic=INTEGER：计算 IRFs 模拟序列数。默认值：order=1 时为 1，否则为 50。

simul_replic=INTEGER：要求经验矩时模拟的序列数（即 periods>0）。注意，如果选项大于 1，附加序列不会用于计算经验矩，而只会以二进制形式保存到位于文件夹 FILENAME/Output 的文件 FILENAME_simul。默认值：1。

solve_algo=INTEGER：可能的值及其含义参见 *solve_algo*。

aim_solver：使用 Anderson-Moore 算法（AIM）计算决策规则，而不是使用基于广义 Schur 分解的 Dynare 默认方法，仅对一阶近似有效。更多详细信息参见 AIM 网站。

conditional_variance_decomposition=INTEGER

conditional_variance_decomposition=[INTEGER1:INTEGER2]

conditional_variance_decomposition=[INTEGER1 INTEGER2...]：计算指定时期的条件方差分解。时期必须严格为正，$var(y_{t+k}|t)$给出条件方差。对于首期，条件方差分解提供了冲击影响效应的分解。结果存储在 oo_.conditional_variance_decomposition（参见 *oo_.conditional_variance_decomposition*）。存在测量误差的情况下，oo_.conditional_variance_decomposition 字段将包含去除测量误差后的方差贡献，即分解实际变量而不是测量变量。测量变量的方差分解将存储在 oo_.conditional_variance_decomposition_ME 中（参见 *oo_.conditional_variance_decomposition_ME*）。方差分解适用于仅在需要理论矩的情况下，即使用 periods=0 选项。仅适用于 order<3 且没有 pruning 选项的情况。在 order=2 的情况下，Dynare 基于二阶解的线性项提供对真实二阶矩的精确近似（参见 Kim、Kim、Schaumburg 和 Sims，2008）。注意，如果需要理论矩，也没有设置 nodecomposition（参见 *oo_.variance_decomposition*），则无条件方差分解（即在水平无穷大处）会自动进行。

pruning：迭代计算模拟解时，放弃高阶项。二阶近似时，Dynare 使用 Kim、Kim、Schaumburg 和 Sims（2008）的算法，三阶近似时，使用 Andreasen、Fernández-Villaverde 和 Rubio-Ramˊırez（2018）的一般化算法。高于三阶不可行。特别指定时，理论矩是基于修剪后的状态空间，即二阶矩的计算使用了 Andreasen、Fernández-Villaverde 和 Rubio-Ramírez（2018）第 10 页中的所有项目，而不是像 Kim、Kim、Schaumburg 和 Sims（2008）那样简单地提供一个基于线性解的二阶精确结果。

partial_information：按照 Pearlman、Currie 和 Levine（1986）在部分信息情形下求解模型。代理人只观察得到经济的部分变量。使用 varobs 命令声明观察到的变量集。注意，如果 varobs 不存在或包含所有内生变量，那就是完全信息情况，此选项无效。更多参考资料参见 Dynare 网站。

sylvester=OPTION：确定求解模块分解模型的 Sylvester 方程的算法。选项可能是：

default：使用基于 Ondra Kamenik 算法的默认解法来求解 Sylvester 方程（gensylv）（参见 Dynare 网站了解更多信息）。

fixed_point：采用不动点算法求解 Sylvester 方程（gensylv_fp）。这种方法对于大型模型来说比默认方法快。

默认值：default。

sylvester_fixed_point_tol=DOUBLE：不动点解的收敛准则。默认值：1e-12。

dr=OPTION：确定计算决策规则的方法。选项可能是：

default：使用基于广义 Schur 分解的默认方法计算决策规则（更多信息参见 Villemot，2011）。

cycle_reduction：利用循环简化算法求解多项式方程，检索决策规则中与内生变量关联的系数。这种方法在大型模型中比默认方法快。

logarithmic_reduction：利用对数简化算法求解多项式方程，检索决策规则中与内生变量关联的系数。这种方法通常比 cycle_reduction 慢。

默认值：default。

dr_cycle_reduction_tol=DOUBLE：循环简化算法的收敛准则。默认值：1e-7。

dr_logarithmic_reduction_tol=DOUBLE：对数简化算法的收敛准则。默认值：1e-12。

dr_logarithmic_reduction_maxiter=INTEGER：对数简化算法使用的最大迭代次数。默认值：100。

loglinear：参见 *loglinear*。注意，所有变量都是通过使用雅可比变换取对数，而不仅仅是选中的变量。因此，必须确保变量有严格的正稳态。stoch_simul 显示对数线性化变量的矩、决策规则和脉冲响应。在 oo_.dr 中保存的决策规则以及模拟变量提到的也是对数线性变量。

tex：请求在 TeX 表和图形中输出结果和图形，稍后可以直接包含在 LaTeX 文件中。

dr_display_tol=DOUBLE：决策规则显示的精度。所有小于 dr_display_tol 的项都不显示在行中。默认值：1e-6。

contemporaneous_correlation：在 oo_.contemporaneous_correlation 中保存内生变量之间的同期相关性。要求不设置 nocorr 选项。

spectral_density：触发计算并显示（滤波）模型变量的理论谱密度。结果存储在下文定义的 oo_.SpectralDensity。默认值：不请求谱密度估计。

hp_ngrid=INTEGER：已弃用的选项。具有与 filtered_theoretical_moments_grid 相同的效果。

输出

这个命令整合了 oo_.dr、oo_.mean、oo_.var、oo_.var_list 和 oo_.autocorr，如下所述。

如果存在 periods 选项，设置 oo_.skewness、oo_.kurtosis 和 oo_.endo_simul（参见 *oo_.endo_simul*），模拟变量保存在全局工作空间的 MATLAB/Octave 向量，名称与内生变量相同。

如果选项 irf 不等于 0，设定 oo_.irfs（参见下文），保存 IRFs 在全局工作空间 MATLAB/Octave 的向量中（后一种访问 IRFs 的方法现已废弃，未来的版本中会消失）

如果选项 contemporaneous_correlation 不等于 0，设置 oo_.contemporaneous_correlation，如下所述。

示例

```
shocks;
var e;
stderr 0.0348;
end;


stoch_simul;
```

执行只有一个随机冲击 e 的二阶近似模拟，标准误为 0.0348。

示例

```
stoch_simul(irf=60) y k;
```

模拟模型，显示变量 y 和 k 的 60 期脉冲响应函数。

MATLAB/Octave variable:oo_.mean

运行 stoch_simul 之后，文件包含内生变量的均值。如果没有 periods 选项，包含理论均值，否则就是模拟均值。变量按照声明顺序排列。

MATLAB/Octave variable:oo_.var

运行 stoch_simul 之后，文件包含内生变量的方差－协方差。如果没有 periods 选项，包含理论方差，否则包含模拟方差。变量按照声明顺序排列。只适用于 order<4。order=2 时将是一个二阶精确近似（即忽略了使用完整的二阶策略函数时产生的三阶项和四阶项）。order=3 时，理论矩只有使用 pruning 才能得到。变量按照声明顺序排列。

MATLAB/Octave variable:oo_.var_list

显示结果的变量列表。

MATLAB/Octave variable:oo_.skewness

运行 stoch_simul 之后，如果存在 periods 选项，文件包含了模拟变量的偏度（标准化三阶矩）。变量按照声明顺序排列。

MATLAB/Octave variable:oo_.kurtosis

运行 stoch_simul 之后，如果存在 periods 选项，文件包含了模拟变量的峰度（标准化的四阶矩）。变量按照声明顺序排列。

MATLAB/Octave variable:oo_.autocorr

运行 stoch_simul 之后，文件包含内生变量的自相关矩阵的元胞数组。元胞数组中矩阵的元素序号对应自相关顺序。选项 ar 指定可用的自相关矩阵数量。如果没有 periods 选项，包含理论自相关系数，否则包含模拟自相关系数。只适用于 order<4。order=2 时将是一个二阶精确近似。order=3 时，理论矩只有使用 pruning 才能得到。只有在静态变量存在的情况下才会创建域。

元素 oo_.autocorr{i}(k,l)等于y_t^k和y_{t-i}^l之间的相关系数。其中，y^k（y^l）是声明顺序中第k（l）个内生变量。

注意，如果需要理论矩，oo_.autocorr{i}等于 oo_.gamma_y{i+1}。

MATLAB/Octave variable:oo_.gamma_y

运行 stoch_simul 之后，如果需要理论矩（即如果没有 period 选项），此文件包含具有以下值的元胞数组（其中 ar 为同名选项的值）：

oo_.gamma{1}：方差/协方差矩阵。

oo_.gamma{i+1}（for i=1:ar）：自相关函数。参见 *oo_.autocorr*。注意，这是自相关函数，不是自协方差函数。

oo_.gamma{ar+2}：无条件方差分解，参见 *oo_.variance_decomposition*

oo_.gamma{ar+3}：如果要求二阶近似，包含均值修正项向量。只适用于 order<4。order=2 时，理论二阶矩是真实二阶矩二阶精确近似。参见 *conditional_variance_decomposition*。order=3 时，理论矩只有使用 pruning 才能得到。

MATLAB/Octave variable:oo_.variance_decomposition

请求理论矩（periods=0）时运行 stoch_simul 之后，包含具有无条件方差分解的结果（即在水平无穷远处）矩阵：第一维度对应内生变量（按命令后或 M_.endo_names 声明的顺序），第二维度对应外生变量（按声明顺序）。数字以百分比表示，各列的总和为 100。存在测量误差的情况下，该字段将包含去除测量误差后的方差贡献，即分解实际变量而不是测量变量。

MATLAB/Octave variable:oo_.variance_decomposition_ME

如果存在测量误差，则请求理论矩（periods=0）时在运行 stoch_simul 后设置字段，类似于 *oo_.variance_decomposition*，但分解是对测量变量。该字段包含带有无条件方差分解（即在水平无穷远处）结果的矩阵：第一维度对应观察到的内生变量（按命令后声明的顺序），第二维度对应外生变量（按声明的顺序），第三维度对应测量误差的贡献。数字以百分比表示，各列的总和为 100。

MATLAB/Octave variable:oo_.conditional_variance_decomposition

使用 conditional_variance_decomposition 选项运行 stoch_simul 之后，包含带有分解结果的三维数组：第一维度对应内生变量（按照命令后的声明顺序，如未指定按照 M_.endo_names 顺序），第二维度对应预测范围（按选项声明），第三维度对应外生变量（按声明顺序）。存在测量误差情况下，该字段将包含去除测量误差后的方差贡献，即分解实际变量而不是测量变量。

MATLAB/Octave variable:oo_.conditional_variance_decomposition_ME

如果存在测量误差，使用 conditional_variance_decomposition 选项运行 stoch_simul 后设置字段，类似于 *oo_.conditional_variance_decomposition*，但分解对被测变量。它包含带有分解结果的三维数组：第一维度对应内生变量（按照命令后的声明顺序，如未指定按照 M_.endo_names 顺序），第二维度对应预测范围（如选项声明），第三个维度对应于外生变量（按声明顺序），第三维度对应测量误差的贡献。

MATLAB/Octave variable:oo_.contemporaneous_correlation

使用 contemporaneous_correlation 选项运行 stoch_simul 之后，如果周期选项不存在，包含理论同期相关性，否则包含模拟同期相关性。只适用于 order<4。order=2 时是二阶精确近似。order=3 时，理论矩只有使用 pruning 才能得到。变量按声明顺序排列。

MATLAB/Octave variable:oo_.SpectralDensity

使用选项 spectal_density 运行 stoch_simul 之后，包含模型变量的谱密度。将有一个 nvars×nfrequencies 大小的子字段 freqs 存储各自的频率网格点，范围从0到2π，以及一个同样大小的子字段 density 存储相应密度。

MATLAB/Octave variable:oo_.irfs

使用不同于零的 irf 选项运行 stoch_simul 之后，包含具有以下命名惯例的脉冲响应：*VARIABLE_NAME_SHOCK_NAME*。例如，oo_.irfs.gnp_ea 包含对 ea 的一个标准差冲击对 gnp 的影响。

MATLAB/Octave command:get_irf('EXOGENOUS_NAME'[,'ENDOGENOUS_NAME']...);

给定外生变量的名称，返回请求的内生变量的 IRFs，因为它们存储在 oo_.irfs。

模型近似解采用一组决策规则或转换方程的形式，决策规则或转换方程表示模型内生变量的当前值作为周期开始时观察到的冲击和模型先前状态的冲函数。决策规则存储在结构 oo_.dr 中，如下所述：

MATLAB/Octave variable:oo_.dr

存储决策规则的结构。下面解释了不同近似阶数的子字段。

Command:extended_path;

Command:extended_path(OPTIONS...);

使用 Fair 和 Taylor（1983）提出的扩展路径法模拟一个随机（即理性预期）模型。内生变量时间序列的产生是假定代理人相信随后的时期不再有冲击。

该函数首先计算外生变量的随机路径（存储在 oo_.exo_simul 中，参见 *oo_.exo_simul*），然后以稳态作为起始点，计算内生变量相应的路径。模拟结果存储在 oo_.endo_simul 中（参见 *oo_.endo_simul*）。注意，此模拟方法不能解出政策和转移方程，而是求解内生变量的路径。

选项

periods=INTEGER：计算模拟的周期数。没有默认值，强制选项。

solver_periods=INTEGER：算法的每次迭代中求解完美预期的周期数。默认值：200。

order=INTEGER：如果 order 大于 0，Dynare 使用高斯正交法解释未来不确定性的影响。如果 order 等于S，内生变量的时间序列的生成是假设代理人认为$t+S$期后不再有冲击。这是一个测试功能，可能会相当慢。非零值与 bytecode 或 model 模块的 block 选项都不兼容。默认值：0。

hybrid：使用二阶扰动常数还原形式修正（随机）扩展路径算法所产生的路径。。

lmmcp：使用 Levenberg-Marquardt 混合互补问题（LMMCP）算法（Kanzow 和 Petra，2004）求解完美预期模型，该算法允许考虑内生变量的不等式约束（例如名义利率的 ZLB 或不可逆投资的模型）。如需指定必要的 mcp 标签，参见 *lmmcp*。

4.13.2 变量类型和顺序

Dynare 区分了四种内生变量：

纯后向（或纯预先决定）变量：仅出现在模型中当前时期和过去时期，但不会出现在未来期（即t和$t-1$但不是$t+1$），数量存储在 M_.npred。

纯前向变量：仅出现在模型中当前时期和未来时期，但不会出现过去期（即t和$t+1$但不是$t-1$），数量存储在 M_.nfwrd。

混合变量：出现在模型当前、过去和未来时期（即t、$t+1$和$t-1$），数量存储在 M_.nboth。

静态变量：仅出现在模型当前时期，而非过去时期和未来时期（即仅出现在t，而不是在$t+1$或$t-1$），数量存储在 M_.nstatic 中。

注意，所有内生变量都属于这四个类别中的一个，因为辅助变量创建后（参见 4.6 辅助变量），所有内生变量至多只有一期超前和一期滞后。因此，我们有以下等式：

```
M_.npred+M_.both+M_.nfwrd+M_.nstatic=M_.endo_nbr
```

MATLAB/Octave variable:M_.state_var

识别声明变量向量中的状态变量的数字索引向量。因此，M_.endo_names（M_.state_var）得到所有在模型声明中属于状态变量的名称，即出现滞后。

Dynare 内部使用两种内生变量的顺序：声明顺序（反映在 M_.endo_names），以及基于上述四种类型的决策规则顺序（“DR”代表决策规则）。大多数情况下使用声明顺序，但决策规则的元素使用决策规则顺序。

决策规则顺序如下：首先出现静态变量，然后是纯后向变量，然后是混合变量，最后是纯前向变量。在每个类别中，变量按照声明顺序排列。

MATLAB/Octave variable:oo_.dr.order_var

将决策规则顺序映射到声明顺序。

MATLAB/Octave variable:oo_.dr.inv_order_var

包含逆映射。换句话说，决策规则顺序中的第k个变量对应于声明顺序中编号为 oo_.dr.order_var(k)的内生变量。相反，第k个声明变量按决策规则顺序编号为 oo_.dr.inv_order_var(k)。

最后，模型的状态变量是纯后向变量和混合变量。当它们出现在决策规则元素中时，按决策规则顺序排列。有 M_.nspred=M_.npred+M_.nboth 这样的变量。类似还有 M_.nsfwrd=M_.nfwrd+M_.nboth，和 M_.ndynamic=M_.nfwrd+M_.nboth+M_.npred。

4.13.3 一阶近似

近似的标准形式：

$$y_t = y^s + \mathrm{A}y_{t-1}^h + \mathrm{B}u_t$$

其中y^s是y的稳态值，$y_t^h = y_t - y^s$。

MATLAB/Octave variable:oo_.dr.state_var

给定当前计算决策规则已经计算的参数值，识别声明变量向量中的状态变量的数字索引向量。可能与 M_.state_var 不同，防止状态变量从给定当前参数化的模型中删除，因为它在决策规则中仅获得 0 个系数。参见 *M_.state_var*。

决策规则的系数存储如下：

- y^s存储在 oo_.dr.ys。向量行对应以声明顺序排列的所有内生变量；
- A存储在 oo_.dr.ghx。矩阵行对应决策规则顺序排列的所有内生变量，列对应决策规则顺序排列的状态变量，由 oo_.dr.state_var 给出（注意：如果已经指定了 model 模块的 block 选项，行是按声明顺序排列，列是根据 oo_.dr.state_var 排列，后者可能与决策规则顺序不同）；
- B存储在 oo_.dr.ghu。矩阵行对应决策规则顺序排列的所有内生变量，列对应声明顺序排列的外生变量（注意：如果已经指定了 model 模块的 block 选项，行对应声明顺序）。

当然，所示近似形式仅仅是形式化的，因为忽略了y^s和y_t^h所需不同阶数。显示 Dynare 处理声明顺序和决策规则顺序之间差异方式的近似精确形式是：

$$y_t(oo\_.dr.order_var)$$
$$= y^s(oo\_.dr.order_var) + A \cdot y_{t-1}(oo\_.dr.order\_.var(k2))$$
$$- y^s\big(oo\_.dr.order_var(k2)\big) + B \cdot u_t$$

其中$k2$选择状态变量，y_t和y^s按声明顺序，系数矩阵按决策规则顺序。实际上，右侧的所有变量都被置于决策规则顺序计算，然后按声明顺序分配给y_t。

4.13.4 二阶近似

近似形式如下：

$$y_t = y^s + 0.5\Delta^2 + Ay_{t-1}^h + Bu_t + 0.5C\big(y_{t-1}^h \otimes y_{t-1}^h\big) + 0.5D(u_t \otimes u_t) + E\big(y_{t-1}^h \otimes u_t\big)$$

其中y^s是y的稳态值，$y_t^h = y_t - y^s$，Δ^2是未来冲击方差的转移效应。因为声明顺序和决策规则顺序的差异而需要重新排序的问题，参见一阶近似。

决策规则的系数存储位置与一阶近似相同，另补充以下储存位置：

- Δ^2存储在 `oo_.dr.ghs2`。向量行对应决策规则顺序排列的所有内生变量；
- C存储在 `oo_.dr.ghxx`。矩阵行对应决策规则顺序排列的所有内生变量，列对应决策规则顺序排列的状态变量向量的 Kronecker 积；
- D存储在 `oo_.dr.ghuu`。矩阵行对应决策规则顺序排列的所有内生变量，列对应声明顺序排列的外生变量的 Kronecker 积；
- E存储在 `oo_.dr.ghxu`。矩阵行对应决策规则顺序排列的所有内生变量，列对应状态变量向量（决策规则顺序）与外生变量向量（声明顺序）的 Kronecker 积。

4.13.5 三阶近似

近似的形式如下：

$$y_t = y^s + G_0 + G_1 z_t + G_2(z_t \otimes z_t) + G_3(z_t \otimes z_t \otimes z_t)$$

其中y^s是y的稳态值，z_t是一个向量，由$t-1$期状态变量（决策规则顺序）和t期外生变量（声明顺序）对稳态的偏离组成。因此，向量z_t的大小为n_z=`M_.nspred+M_.exo_nbr`。

决策规则的系数存储如下：

- y^s存储在 `oo_.dr.ys`。向量行对应声明顺序排列的所有内生变量；
- G_0存储在 `oo_.dr.g_0`。向量行对应决策规则顺序排列的所有内生变量；
- G_1存储在 `oo_.dr.g_1`。矩阵行对应决策规则顺序排列的所有内生变量，列对应决策规则顺序排列的状态变量，后面是声明顺序排列的外生变量；
- G_2存储在 `oo_.dr.g_2`。矩阵行对应决策规则顺序排列的所有内生变量，列对应状态变量（决策规则顺序）的 Kronecker 积，随后是外生变量（声明顺序）。注意，Kronecker 积以折叠方式存储，即对称元素仅存储一次，这意味着矩阵具有$n_z(n_z+1)/2$列。更确切地说，矩阵每列对应于一对（i_1，i_2），其中每个索引表示z_t的元素，因此i在1和n_z之间。只存储非递减元素对，即$i_1 \le i_2$。列按照非递

减对的字母顺序排列。另请注意，对于$i_1 \neq i_2$的元素对，由于元素仅存储一次但在展开的G_2矩阵中出现两次，因此在计算决策规则时必须乘以 2；

- G_3存储在 oo_.dr.g_3。矩阵行对应决策规则顺序排列的所有内生变量，列对应状态变量（决策规则顺序）的三次 Kronecker 幂，随后是外生变量（声明顺序）。注意，三次 Kronecker 幂以折叠方式存储，即对称元素仅存储一次，这意味着矩阵具有$n_z(n_z+1)(n_z+2)/6$列。更确切地说，矩阵每列对应于元组（i_1，i_2，i_3），其中每个索引表示z_t的元素，因此i在 1 和n_z之间。只存储非递减元组，即$i_1 \leq i_2 \leq i_3$。列按照非递减元组的字母顺序排列。另请注意，对于具有三个不同索引的元组（即$i_1 \neq i_2$且$i_1 \neq i_3$且$i_2 \neq i_3$），由于这些元素仅存储一次但在展开的G_3矩阵中出现六次，因此计算决策规则时必须乘以 6。类似地，对于那些具有两个相等索引的元组（即形式为（a,a,b）或（a,b,a）或（b,a,a）），由于这些元素仅存储一次但在展开的G_3矩阵中出现三次，因此在计算决策规则时必须乘以 3。

4.13.6 高阶近似

高阶近似只是对在三阶近似章节中包含内容的一般化。

稳态存储在 oo_.dr.ys 中，常数修正存储在 oo_.dr.g_0 中。1、2、3、4……阶的系数分别存储在 oo_.dr.g_0、oo_.dr.g_1、oo_.dr.g_2、oo_.dr.g_3、oo_.dr.g_4……。这些矩阵的列对应状态变量的多维索引，这种情况下对称元素永远不会重复（详情参见三阶情况下 oo_.dr.g_3 的描述）。

4.14 偶然受限约束（OCCBIN）

Dynare 可以采用 Guerrieri 和 Iacoviello（2015）提出的分段线性解决方法，模拟具有至多两个偶然受限约束的模型，允许使用 Cuba-Borda、Guerrieri、Iacoviello 和 Zhong（2019）提出的反演滤波器或 Giovannini、Pfeiffer 和 Ratto（2021）提出的分段卡尔曼滤波器估计。触发涉及偶然受限约束的计算，需要：

1. 使用occbin_constraints模块定义和命名偶然受限约束；
2. 使用适当的方程标签为model模块中的各个状态指定模型方程；
3. 使用shocks(surprise)模块潜在地指定一系列意外冲击；
4. 使用occbin_setup设置偶然受限约束的模拟或估计；
5. 使用occbin_solver或运行estimation或calib_smoother触发模拟。

下面将讨论所有上述元素：

Block:occbin_constraints;

occbin_constraints模块指定偶然受限约束，包含以下一行或两行：

name‘STRING’;bindEXPRESSION;[relaxEXPRESSION;][error_bindEXPRESSION;][error_relaxEXPRESSION;]

STRING特指relax/bind方程标签中引用约束条件的名称，用来识别各自的状态

（见下文）。bind表达式是必选的，定义了在基线（非约束）状态中评估的逻辑条件，以检查约束是否变为约束。相比之下，relax表达式是可选的，指定了在约束状态中评估的逻辑条件，以检查该机制是否返回到基线非约束状态。如果未指定，Dynare将简单地在约束状态中检查bind表达式计算结果的真假。但是，某些情况无法在约束状态评估bind表达式，因为所涉及的变量根据定义是恒定的，例如需要检查互补松弛条件的拉格朗日乘子。此时有必要提供一个明确的条件，可以在约束状态评估，以检查是否应该保留。

指定表达式时要牢记三件事：第一，可行表达式可能只包含同时期的内生变量。如果要包括超前/滞后变量或外生变量，则需要定义一个辅助变量；第二，Dynare在当前阶段不会线性近似输入的表达式。因为Occbin将使用线性化模型，所以一致性通常需要用户输入线性化约束。否则，检查约束违反的条件可能与基于分段线性模型解决方案的模型模拟使用的条件不同；第三，与最初的Occbin复制代码相比，表达式中使用的变量不会自动减掉平均数，即它们指的是水平值，而不是与稳态的偏差。要访问变量的稳态水平值，可以使用STEADY_STATE()运算符。

error_bind和error_relax为可选项，允许指定数值计算使用的各自约束违反的数字大小标准。默认情况下，Dynare将简单地使用bind和relax不等式的绝对值。但是有些情况下，用户指定的表达式表现更好。

示例

```
occbin_constraints;
    name 'IRR'; bind log_Invest-log(steady_state(Invest))<log
(phi);relax Lambda<0;
    name 'INEG'; bind log_Invest-log(steady_state(Invest))<0;
end;
```

IRR对应不可逆投资的约束，如果在非绑定状态下投资低于稳态的0.025%，则该投资具有约束力。只要约束状态下相应的拉格朗日乘子Lambda变为负时，约束就会放松。注意，此处的约束采用线性形式以便与分段线性模型解决方案保持一致

各个状态和模型方程之间的指定在model模块完成，方程标签用于指示特定方程属于哪个状态。跨状态的所有方程式都必须附有name标签，以便标识同一方程在不同状态下的不同版本。然后将指定的约束名称与bind或relax标签结合使用，以指示特定方程属于哪个状态。在存在多个偶然受限约束的情况下，如果一个方程属于多个状态（例如，两个受限约束），则约束名称标签必须用逗号分隔。如果仅存在一个名称标签，则假定相应的方程对于另一个约束的两种状态都成立。

示例

```
[name='investment',bind='IRR,INEG']
(log_Invest - log(phi*steady_state(Invest)))=0;
```

```
[name='investment',relax='IRR']
Lambda=0;
[name='investment',bind='IRR',relax='INEG']
(log_Invest - log(phi*steady_state(Invest)))=0;
```

投资条件的三个输入方程定义了两个约束条件的所有四个可能组合的模型方程：第一个定义了IRR和INEG都有约束力的模型方程；第二个定义了IRR无约束力的模型方程，无论INEG是否有约束力；最后一个方程定义了最终状态的模型方程，其中IRR有约束力，但INEG没有。

```
Block:shocks(surprise);
Block:shocks(surprise,overwrite);
```

shocks(surprise)模块允许指定外生变量值的一系列临时变化，这些变化在每个时期都会让代理人出乎意料。注意，在occbin_solver等后续命令中使用指定的冲击，需要在该模块后调用occbin_setup。

该模块反映了完美预期的语法，因为它包含一次或多次以下三行代码组：

```
var VARIABLE_NAME;
periods INTEGER[:INTEGER] [[,] INTEGER[:INTEGER]]...;
values DOUBLE | (EXPRESSION)  [[,] DOUBLE | (EXPRESSION)
 ]...;
```

示例（带有向量值和覆盖选项）

```
shockssequence=randn(100,1)*0.02;


shocks(surprise,overwrite);
var epsilon;
periods 1:100;
values (shockssequence);
end;
```

```
Command:occbin_setup;
Command:occbin_setup(OPTIONS...);
```

准备有偶然受限约束的模拟，还翻译shocks(surprise)模块的内容用于后续命令。

为了对偶然受限约束使用estimation，需要调用occbin_setup触发使用反演滤波器或分段卡尔曼滤波器（默认）。在估计的背景下可能出现的问题是结构性冲击在特定状态下从模型中退出。例如，在利率下限为零时，泰勒规则的货币政策冲击将不再出现。如果可观察量多于冲击，可能会产生随机奇点问题。为避免此问题，零利率的数据点应设为NaN，并在使用heteroskedastic_shocks模块时将对应时期的关联冲击标准差设为0。

示例

```
occbin_setup(likelihood_inversion_filter,smoother_inversion_f
ilter);
estimation(smoother,heteroskedastic_filter,...);
```

上述的代码设置了一个在似然评估和平滑器中使用反演滤波器的例子，同时还使用heteroskedastic_filter选项考虑heteroskedastic_shocks。

注意，Occbin的选项大多数都是特定命令指定的，比如当平滑器调用或计算似然值时，有单独的选项来控制Occbin的行为。后面的这些命令不会继承先前为模拟设置的选项。

选项

simul_periods=INTEGER：模拟的时间期数。默认值：100。

simul_maxit=INTEGER：尝试找到分段解方案时的最大迭代次数。默认值：30。

simul_check_ahead_periods=INTEGER：提前检查以恢复到基线状态的周期数。这个数字应该足够大，因为Occbin要求模拟在时间结束时返回到基线状态。默认值：200。

simul_curb_retrench：一次仅更新一期的猜测，而不是将对当前迭代初始状态的猜测建立在上次迭代的基础上。虽然减慢迭代速度，但可能会有更稳健的收敛行为。默认值：未启用。

simul_periodic_solution：接收一个在迭代的两组结果之间交替的周期性解决方案，即结果不唯一。默认值：未启用。

simul_debug：求解过程中提供额外的调试信息。默认值：未启用。

smoother_periods=INTEGER：模拟期间，当平滑器调用时的周期数（等价于simul_periods）。默认值：100。

smoother_maxit=INTEGER：模拟期间，当平滑器调用时使用的最大迭代次数（等价于simul_maxit）。默认值：30。

smoother_check_ahead_periods=INTEGER：模拟期间，当平滑器调用时提前检查以返回基线状态的周期数（等价于simul_check_ahead_periods）。默认值：200。

smoother_curb_retrench：模拟期间，平滑器调用simul_curb_retrench选项。默认值：未启用。

smoother_periodic_solution：接收在两组结果之间交替的周期性解决方案（等价于simul_periodic_solution）。默认值：未启用。

likelihood_periods=INTEGER：模拟期间，计算似然值时使用的周期数（等价于simul_periods）。默认值：100。

likelihood_maxit=INTEGER：模拟期间，计算似然值时使用的最大迭代次数（等价于simul_maxit）。默认值：30。

likelihood_check_ahead_periods=INTEGER：计算似然值时，模拟期间要提前检查以返回基线状态的周期数（等价于simul_check_ahead_periods）。默认值：2

00。

likelihood_curb_retrench：模拟期间，计算似然值调用simul_curb_retrench选项。默认值：未启用。

likelihood_periodic_solution：接受在两组结果之间交替的周期性解决方案（等价于simul_periodic_solution）。默认值：未启用。

likelihood_inversion_filter：估计模型时使用Cuba-Borda、Guerrieri、Iacoviello和Zhong（2019）的反演滤波器。默认值：未启用。

likelihood_piecewise_kalman_filter：估计模型时使用Giovannini、Pfeiffer和Ratto（2021）的分段卡尔曼滤波器。默认值：启用。

likelihood_max_kalman_iterations：分段卡尔曼滤波器外循环的最大迭代次数。默认值：10。

smoother_inversion_filter：运行平滑器时，使用Cuba-Borda、Guerrieri、Iacoviello和Zhong（2019）的反演滤波器。基本假设是系统从稳定状态开始。此时反演滤波器将提供所需的平滑器的输出。默认值：未启用。

smoother_piecewise_kalman_filter：运行平滑器时，使用Giovannini、Pfeiffer和Ratto（2021）的分段卡尔曼滤波器。默认值：启用。

filter_use_relaxation：在分段卡尔曼滤波的更新步骤，使用的假设论证算法中触发松弛条件。当新旧猜测状态差异很大时，使用更接近先前猜测的新猜测。在多种解决方案的情况下，往往会提供具有更短持续时间的偶然受限状态（通常更可取）。指定此选项可能会减慢收敛速度。默认值：未启用。

输出

外生变量的路径存储在options_.occbin.simul.SHOCKS.

Command:occbin_solver;

Command:occbin_solver(OPTIONS...);

基于分段线性求解计算具有偶然受限约束的模拟。注意，必须在此命令之前调用occbin_setup，以便模拟考虑先前的shock（surprise）命令。

选项

simul_periods=INTEGER：参见*simul_periods*。

simul_maxit=INTEGER：参见*simul_maxit*。

simul_check_ahead_periods=INTEGER：参见*simul_check_ahead_periods*。

simul_curb_retrench：参见*simul_curb_retrench*。

simul_debug：参见*simul_debug*。

输出

将各种对象输出到oo_.occbin。

MATLAB/Octave variable:oo_.occbin.simul.piecewise

存储了基于分段线性解模拟的矩阵。变量按声明顺序逐列排列（类似于M_.endo_names），而行对应于simul_periods。

MATLAB/Octave variable:oo_.occbin.simul.linear

存储了基于线性解模拟的矩阵，即忽略偶然受限约束。变量按声明顺序逐列排列（类似于M_.endo_names），而行对应于simul_periods。

MATLAB/Octave variable:oo_.occbin.simul.shocks_sequence

存储了模拟过程中使用冲击序列的矩阵。冲击是逐列排列的，按照它们在M_.exo_names中的顺序存储在oo_.occbin.exo_pos。这些行对应shock（surprise）模块指定的冲击周期数，可能小于simul_periods。

MATLAB/Octave variable:oo_.occbin.simul.regime_history

存储了状态历史信息的结构，以相应时期发生的冲击为条件（沿行存储）。type等于smoother或simul，具体取决于输出是来自模拟运行还是平滑器。子字段regime包含存储状态的向量，而regimestart表示相应状态初始的预期。例如，如果第40行包含针对regime2的[1,0]和针对regimestart2的[1,6]，这表-在第40期的冲击发生之后，第二个约束变得具有约束力(1)，有望在包括当前一个时期在内的六个时期之后（即第45期）后恢复为不具有约束力(0)。

MATLAB/Octave variable:oo_.occbin.simul.ys

稳态值向量。

Command:occbin_graph[VARIABLE_NAME...];

Command:occbin_graph(OPTIONS...)[VARIABLE_NAME...];

绘制图表，比较分段线性解的模拟结果与偶然受限约束中忽略约束的线性解。

选项

Noconstant：省略图表中的稳态。

Command:occbin_write_regimes;

Command:occbin_write_regimes(OPTIONS...);

将记录在oo_.occbin.simul.regime_history或``oo_.occbin.smoother.regime_history的状态历史信息写入储存在FILENAME/Output文件夹的Excel文件上。

选项

periods=INTEGER：写入预期状态持续时间的期间数。默认值：写入所有可用期间。

filename=FILENAME：Excel文件名。默认值：FILENAME_occbin_regimes。

simul：从上次运行的模拟中选择状态历史。默认值：启用。

smoother：从平滑器的最后一次运行中选择状态历史。默认设置：使用simul。

4.15 似然估计方法

假设您拥有某些内生变量的观测值，就可以利用 Dynare 估计模型中的某些或者全部参数。Dynare 提供了最大似然估计（Ireland，2004）和贝叶斯技术（Fernández Villaverde 和 Rubio-Ramírez，2004；Rabanal 和 Rubio-Ramirez，2003；Schorfheide，2000；Smets 和 Wouters，2003）。利用贝叶斯方法可以估计 DSGE 模型、VAR 模型或者两者结合的 DSGE-VAR 模型。为了避免随机奇异，观测变量至少与模型的冲击或测量误差一样多。使用一阶近似的估计可以从模型模块分解中获益（参见 *block*）

Command:varobs VARIABLE_NAME...;

声明估计过程中观察的内生变量，必须在数据文件中可用（参见 *estimation_cmd*）。这个命令还可以与 stoch_simul 中的 partial_information 选项配合使用，用于在部分信息情况下声明观测变量集。

模型文件中只允许有一个 varobs。如果在循环中语句中声明观测变量，则可以使用宏处理程序，如下面的第二个例子所示。

示例

```
varobs C y rr;
```

声明内生变量 C、y 和 rr 为观测变量。

示例（带有宏处理器的循环）

```
varobs
@#for co in countries
GDP_@{co}
@#endfor
;
```

Block:observation_trends;

声明观测变量的线性趋势，且线性趋势为模型参数的函数。如果使用 loglinear 选项，它对应观测变量对数形式的线性趋势，即观测值水平值的指数趋势。每行命令都应该是下列形式：

```
VARIABLE_NAME(EXPRESSION);
```

在大多数情况下，在使用 observation_trends 时，变量不应该居中。

示例

```
observation_trends;
Y (eta);
P (mu/eta);
end;
```

Block:estimated_params;

列出所有待估的参数，必要时还需要指定参数上下限以及先验分布。每行命令对应一

个待估参数。

极大似然估计中，每行遵循以下语法：

```
stderr VARIABLE_NAME|corr VARIABLE_NAME_1,VARIABLE_NAME_2|PAR
AMETER_NAME,INITIAL_VALUE[,LOWER_BOUND,UPPER_BOUND];
```

在贝叶斯 MCMC 估计或者带惩罚项的矩估计，每行遵循以下语法：

```
stderr VARIABLE_NAME|corr VARIABLE_NAME_1,VARIABLE_NAME_2|PAR
AMETER_NAME|DSGE_PRIOR_WEIGHT[,INITIAL_VALUE[,LOWER_BOUND,UPP
ER_BOUND]],PRIOR_SHAPE,PRIOR_MEAN,PRIOR_STANDARD_ERROR[,PRIOR
_3RD_PARAMETER[,PRIOR_4TH_PARAMETER[,SCALE_PARAMETER]]];
```

每行命令的第一部分包括以下四种选项：

- `stderr VARIABLE_NAME`：外生变量 VARIABLE_NAME 的标准误，或与内生观测变量 VARIABLE_NAME 关联的观测误差/测量误差的标准差，这些标准误是待估量；
- `corr VARIABLE_NAME1,VARIABLE_NAME2`：外生变量 VARIABLE_NAME1 和 VARIABLE_NAME2 间的相关系数，或与内生观测变量 VARIABLE_NAME1 和 VARIABLE_NAME2 关联的观测误差/测量误差的相关系数，这些相关系数是待估量。注意，如果在前面的 `shocks` 模块或 `estimation` 命令中设定的相关系数随后不再估计，在保持估计前设定的值。因此，处理方法与模型校准时设置深度参数的方法相同，无需估计；
- `PARAMETER_NAME`：待估计的模型参数的名称；
- `DSGE_PRIOR_WEIGHT`：DSGE-VAR 模型中 DSGE 模型权重的特殊名称。

其余部分由以下字段组成，其中一些是可选的：

`INITIAL_VALUE`：指定后验模式最优算法或极大似然估计的初始值。如果未设置，默认为先验均值。

`LOWER_BOUND`：极大似然估计中指定参数值下限。贝叶斯估计中，最大化后验核时设置一个有效下界。下界没有修改先验密度函数的形状，旨在帮助最优算法识别后验模式（对 MCMC 没有影响）。对于某些先验密度函数（即逆伽马、伽马、均匀分布、贝塔分布或韦伯分布），使用 *`prior_3rd_parameter`* 将先验分布的峰值转移到左侧或者右侧。此时可以有效地修改先验密度函数（注意 Dynare 尚不能实施截断高斯密度函数）。如果未设置，默认为负无穷大（极大似然）或先验分布（贝叶斯估计）的自然下界。

`UPPER_BOUND`：和 `LOWER_BOUND` 相同，但是下界改为上界。

`PRIOR_SHAPE`：指定先验密度函数形状的关键词，可能的值是：`beta_pdf`、`gamma_pdf`、`normal_pdf`、`uniform_pdf`、`inv_gamma_pdf`、`inv_gamma1_pdf`、`inv_gamma2_pdf` 和 `weibull_pdf`，注意，`inv_gamma_pdf` 和 `inv_gamma1_pdf` 等价。

PRIOR_MEAN：先验分布的均值。

PRIOR_STANDARD_ERROR：先验分布的标准差。

PRIOR_3RD_PARAMETER：先验分布的第三个参数，用于广义贝塔分布、广义伽马分布、广义韦伯分布和均匀分布。默认值：0。

PRIOR_4TH_PARAMETER：先验分布的第四个参数，用于广义贝塔分布、均匀分布。默认值：1。

SCALE_PARAMETER：声明尺度参数，针对 Metropolis-Hasting 算法的跳跃分布协方差矩阵。

需要注意的是，INITIAL_VALUE、LOWER_BOUND、UPPER_BOUND、PRIOR_MEAN、PRIOR_STANDARD_ERROR、PRIOR_3RD_PARAMETER、PRIOR_4TH_PARAMETER 和 SCALE_PARAMETER 都可以是有效的 EXPRESSION，部分选项可以为空，Dynare 会根据估计方法和先验分布函数选择合适的默认值。

均匀分布情况下，可以使用 *PRIOR_3RD_PARAMETER* 和 *PRIOR_4TH_PARAMETER* 提供上限和下限或使用 *PRIOR_MEAN*、*PRIOR_STANDARD_ERROR* 的平均值和标准差指定，另外两个会自动补充。注意，提供两组超参数将产生错误消息。

命令越靠近列表末尾，前面的选项都要声明，例如：如果声明 SCALE_PARAMETER 选项，就必须先定义 PRIOR_3RD_PARAMETER 和 PRIOR_4TH_PARAMETER，如果这些参数没有给出，则定义为空值。

示例

```
corr eps_1,eps_2,0.5,  ,  ,beta_pdf,0,0.3,-1,1;
```

eps_1 和 eps_2 的相关系数设置为一个均值是 0，方差是 0.3 的广义贝塔分布。通过设置 PRIOR_3RD_PARAMETER 为-1，PRIOR_4TH_PARAMETER 为 1，使得区间为[0,1]的标准贝塔分布转变为区间为[-1,1]的广义贝塔分布。注意，此时 LOWER_BOUND 和 UPPER_BOUND 设置为空，因此分别默认为-1 和 1，初始值为 0.5。

示例

```
corr eps_1,eps_2,0.5,-0.5,1,beta_pdf,0,0.3,-1,1;
```

设置与前面相同的广义贝塔分布，但是现在使用 LOWER_BOUND 和 UPPER_BOUND 将分布区间截断为[-0.5,1]。

参数变换

有时待估模型参数的一个变换参数，而不是模型参数本身。当然，在模型的任何地方，用待估参数的函数代替原始参数都是可能的，但是常常不现实的。在这种情况下，可以在 parameters 语句声明待估参数，并定义参数转换，这一过程是使用#表达式实现（参见 4.5 模型声明）。

示例

```
parameters bet;


model;
#sig=1/bet;
c=sig*c(+1)*mpk;
end;


estimated_params;
bet,normal_pdf,1,0.05;
end;
```

Block:estimated_params_init;

Block:estimated_params_init(OPTIONS...);

声明最优化算法的初始值，当这些值与先验均值不同时。应在 estimated_params 模块之后声明，否则声明的初始值会被后者覆盖。

每行命令都有如下的语句结构：

```
stderr VARIABLE_NAME|corr VARIABLE_NAME_1,VARIABLE_NAME_2|PAR
AMETER_NAME,INITIAL_VALUE;
```

选项

use_calibration：对于未特别初始化的参数，使用深度参数和校准中 shocks 模块指定的协方差矩阵的元素作为估计的起始值。对于在校准期间未明确指定或违反先验的 shocks 模块组件，使用先验均值。各种组件的含义和语法参见 *estimated_params*。

Block:estimated_params_bounds;

在极大似然估计中声明参数的下界和上界。每行命令都有如下语句结构：

```
stderr VARIABLE_NAME|corr VARIABLE_NAME_1,VARIABLE_NAME_2|PAR
AMETER_NAME,LOWER_BOUND,UPPER_BOUND;
```

各组件的意义和语法参见 *estimated_params*。

Command:estimation[VARIABLE_NAME...];

Command:estimation(OPTIONS...)[VARIABLE_NAME...];

执行贝叶斯估计和极大似然估计，会展示如下信息：

- 后验优化结果（也包括极大似然估计）；
- 边际对数数据密度；
- 后验均值和来自后验模拟的最高后验密度区间（最小可信集）；
- 如果仅使用一条 MCM 链时，应用收敛诊断表；或者使用多条 MCM 链时，应用 Pfeifer（2014）提出的 Metropolis-Hastings 收敛图；

- MCMC 数值无效因子表；
- 表示先验、后验和众数的图；
- 平滑冲击、平滑观测误差、平滑和历史变量图。

注意，后验矩、平滑变量、k步超前过滤变量和预测（当需要时）仅在 estimation 命令之后列出的变量上计算。或者，可以选择在所有内生变量或所有观察变量上计算相应的结果（参见后文 consider_all_endogenous 和 consider_only_observed 选项）。如果在估计命令之后没有列出变量，Dynare 将交互询问使用哪个变量集。

此外，在 MCMC 期间（mh_replic>0 的贝叶斯估计），会出现等待窗口（图形或文本），显示蒙特卡洛的进度以及当前的接受率。注意，如果使用 load_mh_file 选项（参见下文），则报告的接受率没有考虑来自之前的 MCMC 数据。文献中有一个经验法则，接受率应该接近三分之一或四分之一。如果不在这个范围，可以停止 MCMC（CTRL-C），并改变 mh_jscale 选项的值（参见下文）。

默认情况下，Dynare 使用种子集等于 0 的 mt199937ar 算法（即 Mersenne Twister 方法）生成随机数。因此，Dynare 的 MCMC 是确定性的：在运行不同 Dynare 后（其他条件相同）将得到相同的结果。例如，后验矩或后验密度将完全相同。这个特性允许我们很容易地识别出由于模型、先验数据或估计项的变化带来的结果。但人们可能也想检查一下，多次运行时，不同建议序列中，返回的结果几乎相同。如果迭代次数（即 mh_replic 的值）足够使得 MCMC 收敛到遍历分布，那么应该是正确的。此时不需要默认随机数生成器，用户应该在估计命令之前，根据系统时钟使用以下命令设置种子：

```
set_dynare_seed('clock');
```

这样在不同运行过程中可以得到不同的数据序列。

最后，Dynare 并不能总是正确的区分最大似然法和贝叶斯估计的命令，虽然在频率置信区间和贝叶斯最高后验密度区间（HPDI）之间以及后验密度和概率之间有一个重要的概念区别，但 Dynare 有时使用贝叶斯术语作替代显示的最大似然结果。ML 中 estimation 的 forecast 选项是输出储存的例子，其中用 HPDinf/HPDsup 代表置信区间。

算法

如果 *mh_replic* 的值大于 2000，并且没有使用 *nodiagnostic* 选项，则估计命令会执行 MCMC 诊断。如果 *mh_nblocks* 等于 1，则计算 Geweke（1992，1999）的收敛诊断。舍弃了 *mh_drop* 声明的 burn-in 以后，使用卡方检验比较 *geweke_interval* 确定的首、末次抽取的平均值。检验是利用了不存在序列相关的方差估计和 *taper_steps* 声明的锥形窗口来计算。如果 *mh_nblocks* 大于 1，则使用 Brooks 和 Gelman（1998）的收敛诊断。正如 Brooks 和 Gelman（1998）文献的第三部分所述，单变量收敛诊断是基于集合的比较，也是在 MCMC 矩内的比较（Dynare 显示二阶和三阶矩，“最大概率密度”区间的长度覆盖了 80%的后验分布）。由于计算的原因，多元收敛诊断不严格遵循 Brooks 和 Ge

lman（1998），而是将主要思想应用于后验似然函数的范围，而不是单个参数。后验核用于将参数聚集成标量统计量，然后使用 Brooks 和 Gelman（1998）单变量收敛诊断检查收敛性。

无效因子使用基于 Parzen 窗口（即 Andrews，1991）的 Giordano 等（2011）方法计算。

选项

datafile=FILENAME：数据文件：可以用*.m、*.mat、*.csv 或者*.xls/*.xlsx 文件（Octave 环境下，对于*.csv 和*.xlsx 格式，Octave-Forge 需要安装 io 包，但不支持*.xls）。注意，数据文件的名称（即不包括扩展名）必须不同于模型文件名。如果存在多个名为 FILENAME 的文件，但是具有不同的文件后缀，则文件名必须包含在引号字符串中，并提供如下文件后缀：

```
estimation(datafile='../fsdat_simul.mat',...);
```

dirname=FILENAME：存储 estimation 输出的目录，如果要传递目录的子目录，必须引用参数。默认：<mod_file>。

xls_sheet=NAME：Excel 文件中有数据的 sheet 名称。

xls_range=RANGE：Excel 文件使用的数据范围。例如，xls_range=B2:D200。

nobs=INTEGER：根据 first_obs 使用的观测数据样本数。默认值：文件中所有在 first_obs 之后的观测数据。

nobs=[INTEGER1:INTEGER2]：递归估计和预测从 INTEGER1 到 INTEGER2 的样本量，必须声明 forecast 选项，预测结果存储在 RecursiveForecast 字段中（参见 *RecursiveForecast*）。各自的结果 oo_存储在 oo_recursive_中（参见 *oo_recursive_*），并且用各自的样本长度作为索引。

first_obs=INTEGER：使用的第一个观测变量的样本量，例如，估计 DSGE-VAR 时，first_obs 需要大于滞后变量数。默认值：1。

first_obs=[INTEGER1:INTEGER2]：从 INTEGER1 到 INTEGER2 的第一观测量开始，运行滚动窗口估计和预测固定长度 nobs 的样本，也必须规定 forecast 选项。使用扩展窗口时与递归预测不兼容（参见 *nobs*）。各自的结果 oo_存储在 oo_recursive_中（参见 *oo_recursive_*），并且用各自滚动窗口的首个观测值作为索引。

prefilter=INTEGER：值等于 1 意味着估计程序将会对每个数据序列去除经验均值。如果请求了包括 *logdata* 选项的 *loglinear* 选项，则先取对数，然后去均值。默认值：0，即没有提前滤波。

presample=INTEGER：在评价似然函数前，跳过 first_obs 选项之后观测值的数量。这些预采样观测值不进入似然函数，而是用作执行卡尔曼滤波迭代的样本。此选项与估计 DSGE-VAR 不兼容。默认值：0。

loglinear：计算模型的对数线性近似，而不是线性近似。估计过程中，数据必须对

应模型中使用的变量定义（如何正确地声明连接模型变量和数据的观察方程的信息，参见Pfeifer，2013）。如果声明了 *loglinear* 选项，Dynare 就会采用模型变量和数据的对数，因为假定数据与原始的非对数模型变量对应。所显示的后验结果，如脉冲响应、平滑变量和矩将用于对数变量，而不是原始非对数变量。默认值：计算线性近似。

logdata：如果使用了模型对数线性化选项（*loglinear*），Dynare 对数转换已有数据，除非此时使用 logdata 选项。如果用户提供了已对数化的数据，则需要此选项，否则将应用两次对数转换（这可能导致复杂数据）。

plot_priors=INTEGER：控制先验密度图的绘制：

0：无先验图；

1：绘制每个估计参数的先验密度，重要的是检查先验密度的实际形状是否符合预期。先验标准密度值的不当选择将会导致错误的先验密度。

默认值：1。

nograph：参见 *nograph*。

posterior_nograph：不绘制贝叶斯脉冲响应图（*bayesian_irfs*）、后验平滑结果图（*smoother*）、后验预测图（*forecast*）等。

posterior_graph：重新启用生成先前使用 *posterior_nograph* 关闭的图。

nodisplay：参见 *nodisplay*。

graph_format=FORMAT graph_format=(FORMAT,FORMAT...)：参见 *graph_format*。

no_init_estimation_check_first_obs：首期不检查随机奇异性。如果使用该选项，只在首期检查失败，*ESTIMATION CHECKS* 不会返回错误。当观测到首期存量变量（如资本）在相关流量变量（如投资）之上应该使用这个选项。使用这个选项可能会发生崩溃或者在特定的问题上提供不希望的或者错误的结果（如在首期观测到的附加变量不是预先确定的）。仅供高级场景使用。

lik_init=INTEGER：卡尔曼滤波初始化类型：

1：对于平稳模型，预测误差方差的初始矩阵设置为状态变量的无条件方差；

2：对于非平稳模型，广泛采用的经验法则是，预测误差矩阵对角线上初始方差设置为 10（根据 Harvey 和 Phillips，1979 的建议）；

3：对于非平稳模型，使用扩散滤波（使用 diffuse_filter 选项）；

4：用 Riccati 方程初始化滤波；

5：（1）对非平稳元素使用选项 2，在预测误差矩阵对角线上将初始方差设置为 10，其他所有协方差设置为 0；（2）对平稳元素使用选项 1。

默认值：1，仅用于高级用途。

lik_algo=INTEGER：仅供内部使用和测试。

conf_sig=DOUBLE：估计后用于经典预测的置信区间的显著性水平。默认值：0.9。

mh_conf_sig=DOUBLE：用于计算先验和后验统计的置信/HPD 区间，如：参数分布、先验/后验矩、条件方差分解、脉冲响应函数、贝叶斯预测等。默认值：0.9。

mh_replic=INTEGER：Metropolis-Hastings 算法的迭代次数。抽样数量应该足够充分保证 MCMC 的收敛以及后验分布的计算。默认值：20000。

sub_draws=INTEGER：计算各种对象（平滑变量、平滑冲击、预测、矩、IRF）后验分布的 MCMC 抽样次数。用于计算这些后验矩的抽样是在估计的经验后验分布（即 MCMC 的抽取）中均匀抽取的。sub_draw 应该小于 MCMC 的可用总数。默认值：min(posterior_max_subsample_draws、(Total number of draws)*(number of chains))。

posterior_max_subsample_draws=INTEGER：如果没有通过选项 sub_draw 覆盖，用于计算各种对象的后验分布（平滑变量、平滑冲击、预测、矩、IRF）的 MCMC 的最大抽样次数。默认值：1200。

mh_nblocks=INTEGER：Metropolis-Hastings 算法的平行链数。默认值：2。

mh_drop=DOUBLE：使用后验模拟之前，初始生成的参数向量的一部分作为预迭代而舍弃。默认值：0.5。

mh_jscale=DOUBLE：跳跃分布的协方差矩阵的乘法因子（Metropolis-Hastings 或 TaRB 算法）。默认值总是不令人满意。这个选项必须调整，以获得理想的接受率 25%-33%。基本上，经验法则是如果接受率太高，增加跳跃分布的方差，如果接受率太低，减少跳跃分布的方差。在某些情况下，它可以帮助考虑这个尺度参数的特定值。这些在 *estimated_params* 模块中声明。

注意，mode_compute=6 将调整尺度参数，达到 AcceptanceRateTarget。生成的尺度参数将保存到名为 MODEL_FILENAME_mh_scale.mat 的文件中，这个文件在后续运行中通过 posterior_sampler_options 选项 *scale_file* 加载。mode_compute=6 和 scale_file 都用调整值覆盖 estimated_params 中指定的任何值。默认值：0.2。

另外，对于随机游走 Metropolis-Hastings 算法，可以使用选项 *mh_tune_jscale* 自动调整 mh_jscale 的值。此时禁用 mh_jscale 选项。

mh_init_scale=DOUBLE：用于抽取 Metropolis-Hastings 链初始值的比例参数。一般来说，对于 Brooks 和 Gelman（1998）收敛诊断起始点应该分散开才有意义。默认值：2*mh_jscale。

重要的是，Dynare 开始执行的时候就要设置 mh_init_scale，也就是说默认不考虑由 mode_compute=6 或 posterior_sampler_options 的选项 *scale_file* 引入的 mh_jscale 的潜在变化。如果后验抽样的初始化过程中，mh_init_scale 设置得太宽，以致 100 次测试的抽样不可行（例如，总是违反 Blanchard-Kahn 条件），那么，Dynare 将请求

用户输入一个新的 mh_init_scale 值，然后使用该值进行新一轮的 100 次抽样测试。如果 *nointeractive* 选项已经被调用，在 100 次无效的抽样之后程序会自动减少 10%的 mh_init_scale，并尝试另外 100 次抽样。迭代过程最多进行 10 次，这时 Dynare 会以错误信息中止工作。

mh_tune_jscale[=DOUBLE]：自动调整跳跃分布的协方差矩阵（Metropolis-Hastings）的尺度参数，使整体接受率接近所需水平。默认值为 0.33。由于算法的随机性（如果 mh_nblocks>1，马尔科夫链的建议和初始条件），不可能精确匹配所需的接受率。此选项仅适用于随机游走 Metropolis Hastings 算法，不能与 mh_jscale=DOUBLE 一起使用。

mh_tune_guess=DOUBLE：指定 mh_tune_jscale 选项的初始值。默认值：0.2。如果未使用 mh_tune_jscale，则禁止设置。

mh_recover：从最后一个可用的已保存的 mh 文件开始，试图恢复一个过早崩溃的 Metropolis-Hastings 模拟。该选项不应该与 load_mh_file 一起使用，也不应该采用异于之前崩溃程序的 mh_replic 值。因为 Dynare4.5 会自动加载先前运行保留的提议密度。旧版本为了确保链以相同的提议密度延续下去，应该在使用此选项时提供上次运行中使用的 mode_file 或相同的自定义 mcmc_jumping_covariance。注意，Octave 环境目前不支持用各自的最后一个随机数发生器的状态整齐地延续崩溃的链。

mh_posterior_mode_estimation：跳过基于优化的众数搜索而选择利用 MCMC 来寻找众数。MCMC 在先验分布的众数开始而且使用先验分布的方差计算海塞矩阵的逆。

mode_file=FILENAME：包含上次众数的文件名称。计算众数时，Dynare 将众数（xparam1）和海塞矩阵（hh，仅当 cova_compute=1 时）存储在一个位于 FILENAME/Output 文件夹，名为 MODEL_FILENAME_mod.mat 的文件。成功运行估计命令后，将禁用 mode_file，以防止其他函数隐性调用更新后的众数文件。因此，如果*.mod 文件包含后续的 estimation 命令，则需要再次声明 mode_file 选项。

mode_compute=INTEGER|FUNCTION_NAME：声明众数计算的最优化算法：

0：不计算众数。声明 mode_file 选项时，仅仅从文件中读取众数的值。当没有指定 mode_file 选项时，Dynare 报告在参数初始值处评估的对数后验分布（对数似然函数）；当没有指定 mode_file，且没有 estimated_params 模块时，但是使用 smoother 选项，这是一种迂回的方式来计算参数校准模型的变量的平滑值；

1：使用 fmincon 优化程序（如果安装了优化工具箱，可在 MATLAB 环境使用；如果安装了 Octave-Forge1.6 或更高版本的 optim 组件，可在 Octave 环境使用）；

2：使用 Corana 等（1987）和 Goffe 等（1994）描述的连续模拟退火型全局优化算法。

3：使用 fminunc 优化程序（如果安装了优化工具箱，可在 MATLAB 环境使用；如果安装了来自 Octave-Forge 的 optim 组件，可在 Octave 环境使用）；

4：使用 Chris Sims 的 csminwel；

5：使用 Marco Ratto 的 newrat。这个值与非线性滤波或 DSGE-VAR 模型不兼容。这是片段优化算法（slice optimizer）：大多数迭代是一连串的单变量优化步骤的序列，每个估计的参数或冲击都对应一个。在每个步骤中使用 csminwel 实施线性搜索；

6：使用基于 Monte-Carlo 的优化程序（更多详细信息参见 https://archives.dynare.org/DynareWiki/MonteCarloOptimization）；

7：使用 fminsearch，基于单纯形法的优化程序（如果安装了优化工具箱，可在 MATLAB 环境使用；如果安装了来自 Octave-Forge 的 optim 组件，可在 Octave 环境使用）；

8：使用 Dynare 实现的基于 Nelder-Mead 单纯形法的优化程序（通常比使用 mode_compute=7 的 MATLAB 或 Octave 更高效）；

9：采用 Hansen 和 Kern（2004）的 CMA-ES（协方差矩阵适应演化策略）算法，这是一种用于复杂非线性非凸优化的演化算法；

10：采用基于非线性单纯形法和模拟退火算法的结合的 simpsa 算法，由 Cardoso、Salcedo 和 Feyo de Azevedo（1996）提出；

11：严格地说，这不是一种优化算法。（估计的）参数作为状态变量，与模型原始状态变量一起采用非线性滤波联合估计得到。Dynare 实现的算法在 Liu 和 West（2001）有描述，并与模型的 k 阶局部近似一起工作；

12：使用 particleswarm 优化程序（如果安装了全局优化工具箱，可在 MATLAB 环境使用；不适用于 Octave）；

13：使用 lsqnonlin 非线性最小二乘优化路径（如果安装了优化工具箱，可在 MATLAB 环境使用；如果安装了来自 Octave-Forge 的 optim 组件，可在 Octave 环境使用）。仅支持 method_of_moments；

101：使用 Kuntsevich 和 Kappel（1997）提出的 SolveOpt 算法来解局部非线性优化问题；

102：使用 simulanneelalbnd 优化程序（如果安装了全局优化工具箱，可在 MATLAB 环境使用；不适用于 Octave）；

FUNCTION_NAME：也可以为该选项提供一个 FUNCTION_NAME，而不是 INTEGER。此时 Dynare 将该函数的返回值作为后验众数。

默认值：4。

silent_optimizer：指示 Dynare 后台运行众数计算/优化，而不显示结果或保存文件。运行循环时有用。

mcmc_jumping_covariance=OPTION：告诉 Dynare 在 MCMC 采样器的提议密度中使用哪个协方。。可以是以下选项之一：

hessian：使用众数处计算的海塞矩阵。

prior_variance：使用先验方差。此时不允许有无限的先验方差。

identity_matrix：使用单位矩阵。

FILENAME：从 FILENAME.mat 加载任意用户自定义协方差矩阵。协方差矩阵必须保存在一个名为 jumping_covariance 的变量中，必须是正定方阵，并且与待估参数数目有相同的维数。

注意，协方差矩阵仍然是按 *mh_jscale* 放缩。默认值：hessian。

mode_check：指示 Dynare 依次为每个待估参数在计算所得到的众数附近抽取后验密度函数。这有助于诊断优化算法问题。注意，对于 order>1，由于重采样步骤，粒子滤波器产生的似然函数不再可微。mode_check 图可能因此看起来很不稳定。

mode_check_neighbourhood_size=DOUBLE：与选项 mode_check 结合使用，给出诊断图上显示的后验众数周围的窗口宽度，宽度以百分比偏差表示。允许使用 Inf，将触发整个域的绘图（参见 *mode_check_symmetric_plot*）。默认值：0.5。

mode_check_symmetric_plots=INTEGER：与选项 mode_check 结合使用，如果设置为 1，则告诉 Dynare 确保检查图在后验众数附近是对称的。值 0 允许有非对称图，如果后验众数接近域边界，或者当域不在整条实线上时，与 mode_check_neighbourhood_size=Inf 联合使用，将会很有用。默认值：1。

mode_check_number_of_points=INTEGER：评价（对每个参数）后验核的后验众数附近的点数量。默认值：20。

prior_trunc=DOUBLE：计算参数的边界时，忽略先验密度函数极端值的概率。默认值：1e-32。

huge_number=DOUBLE：当计算原因需要有限值时，在（先验）边界的定义中替换无限值的选项值。默认值：1e7。

load_mh_file：指示 Dynare 添加先前的 Metropolis-Hastings 模拟，而不是从头开始。因为 Dynare4.5 会自动加载先前运行保留的提议密度。旧版本为了确保链以相同的提议密度延续下去，应该在使用此选项时提供上次运行中使用的 mode_file 或相同的自定义 mc mc_jumping_covariance。不应和 mh_recover 一起使用。注意，Octave 环境目前不支持用已经存在的抽取的最后一个随机数发生器的状态整齐地延续链条。

load_results_after_load_mh：当加载先前的 MCMC 运行结果，又不需要添加额外的抽样时，即用 mh_replic=0 指定 load_mh_file，该选项可用。它告诉 Dynare 从现有的_results 文件中加载以前计算过的收敛诊断、边际数据密度和后验统计量，而不是重新计算。

mh_initialize_from_previous_mcmc：允许从以前的 MCMC 挑选初始值，其中模型的规格、估计参数的数量、（一些）先验可能已经改变（在 load_mh_file 不工作的情况下）。如果新增一个额外的待估参数，它将从 prior_draw 中自动初始化。注意，如果使用这个选项跳过优化步骤，应该使用不需要提议密度的抽样方法，比如切片抽样。

否则，优化需要在事前执行或者用一个包含适当的后验协方差矩阵的众数文件。

mh_initialize_from_previous_mcmc_directory=FILENAME：如果设定 mh_initialize_from_previous_mcmc，必须提供标准 FNAME 文件夹路径，从该路径加载先验分布具体定义和最后的 MCMC 值，用于初始化新的 MCMC。

示例：如果前一个项目文件夹在/my_previous_dir 而且 FNAME 是 mymodel，应该设定如下选项：

mh_initialize_from_previous_mcmc_directory='/my_previous_dir/mymodel'。

然后 Dynare 到：

/my_previous_dir/mymodel/metropolis/mymodel_mh_history_<LAST>.mat

寻找上一个记录文件，到：

/my_previous_dir/mymodel/prior/definition.mat

寻找先验分布的定义。

mh_initialize_from_previous_mcmc_record=FILENAME：如果设定 mh_initialize_from_previous_mcmc，而且当不能从目录中找到合适的用来加载初值的文件时，用户可以在此直接提供记录文件的路径，从中加载用于初始化新 MCMC 的值。

mh_initialize_from_previous_mcmc_prior=FILENAME：如果设定 mh_initialize_from_previous_mcmc，而且当不能从目录中找到合适的用来加载初值的文件时，用户可以在此直接提供先验定义文件的路径，获得先前 MCMC 使用的先验信息。

optim=(NAME,VALUE,...)：NAME 和 VALUE 配对的列表。可用于为优化程序设置选项。可用选项的集合取决于所选的优化程序（即取决于选项 mode_compute）：

1、3、7、12、13：MATLAB 优化工具箱的说明文档或 Octave 的说明文档提供了可用选项；

2：可用选项有：

- initial_step_length：初始步长。默认值：1；
- initial_temperature：初始温度。默认值：15；
- MaxIter：函数评估的最大数量。默认值：100000；
- neps：用于决定终止时的最终函数值的数量。默认值：10；
- ns：周期数。默认值：10；
- nt：温度降低前的迭代次数。默认值：10；
- step_length_c：步长调整。默认值：0.1；
- TolFun：停止标准。默认值：1e-8；
- rt：温度下降因子。默认值：0.1；
- verbosity：控制优化过程中显示的冗余，范围从 0（静默）到 3（每个函数评

估）。默认值：1；

4：可用选项有：

- InitialInverseHessian：后验核（或似然函数）的 Hessian 矩阵逆的初始近似值。显然，这个近似必须是一个正定的对称方阵。默认值：1e-4*eye(nx)，其中 nx 为待估参数数量；
- MaxIter：最大迭代次数。默认值：1000；
- NumgradAlgorithm：可能的值分别为 2、3 和 5，分别对应用于计算目标函数梯度的 2、3 和 5 点公式（参见 Abramowitz 和 Stegun，1964）。13 和 15 的值更具有实验性。如果两边的扰动增加了目标函数的值（最小化这个函数），那么就迫使梯度的相应元素为零，思路是暂时减小优化问题的大小。默认值：2；
- NumgradEpsilon：用于计算目标函数梯度的扰动的大小。默认值：1e-6；
- TolFun：停止标准。默认值：1e-7；
- verbosity：控制优化过程中显示的冗余，设置为 0 即静默。默认值：1；
- SaveFiles：优化过程中控制中间结果。设置为 0 即关闭保存。默认值：1。

5：可用选项是：

- Hessian：触发三种类型的海塞矩阵计算。0：外积梯度；1：默认的 Dynare 海塞矩阵程序；2：“混合”外积梯度，其中对角线元素使用二阶导数公式获得，外积用于相关系数结构。{0}和{2}选项都需要单变量滤波，确保使用最大的个体密度和正定海塞矩阵。{0}和{2}都比默认的 Dynare 海塞矩阵程序快，但是为大型模型的 Metropolis 采样算法提供了合适的初值（选项{2}比{0}更精确）。默认值：1；
- MaxIter：最大迭代次数。默认值：1000；
- TolFun：停止标准。默认值：数值导数下为 1e-5，解析导数下为 1e-7；
- verbosity：控制优化过程中显示的冗余，设置为 0 即静默。默认值：1；
- SaveFiles：优化过程中控制中间结果。设置为 0 即关闭保存。默认值：1。

6：可用选项是：

- AcceptanceRateTarget：介于 0 和 1 之间的实数。调整跳跃分布的尺度参数，有效接受率可匹配 AcceptanceRateTarget 选项的值。默认值：1.0/3.0；
- InitialCovarianceMatrix：跳跃分布的初始协方差矩阵。默认值：使用 mode_file 选项为 previous，否则为 prior；
- nclimb-mh：上一个 MCMC（层进式，climbing mode）的迭代次数。默认值：200000；
- ncov-mh：更新跳跃分布协方差矩阵的迭代次数。默认值：20000；
- nscale-mh：调整跳跃分布尺度参数的最大迭代次数。默认值：200000；
- NumberOfMh：依序运行 MCMC 的数量。默认值：3。

8：可用选项是：

- InitialSimplexSize：单纯形法的初始大小，表示在每个方向上与提供的初始猜测值的百分比偏差。默认值：0；
- MaxIter：最大迭代次数。默认值：5000；
- MaxFunEvals：目标函数评估的最大数量，无默认值；
- MaxFunvEvalFactor：设置 MaxFunvEvals=MaxFunvEvalFactor×估计参数的数量。默认值：500；
- TolFun：精度参数（关于目标函数）。默认值：1e-4；
- TolX：精度参数（关于工具）。默认值：1e-4；
- verbosity：控制优化过程中显示的冗余，设置为 0 即静默。默认值：1。

9：可用选项是：

- CMAESResume：恢复先前的运行。需要从上一个运行得到的 variablescmaes.mat，设置为 1 表示启用。默认值：0；
- MaxIter：最大迭代次数；
- MaxFunEvals：目标函数评估的最大数量。默认值：Inf；
- TolFun：精度参数（关于目标函数）。默认值：1e-7；
- TolX：精度参数（关于工具）。默认值：1e-7；
- verbosity：控制优化过程中显示的冗余，设置为 0 即静默。默认值：1；
- SaveFiles：优化过程中控制中间结果。设置为 0 即关闭保存。默认值：1。

10：可用的选项是：

- EndTemperature：温度的终止条件。当温度达到 EndTemperature 时，温度设置为零，算法又回到了标准的单纯形法算法。默认值：0.1；
- MaxIter：最大迭代次数。默认值：5000；
- MaxFunvEvals：目标函数评估的最大数量，无默认值；
- TolFun：精度参数（关于目标函数）。默认值：1e-4；
- TolX：精度参数（关于工具）。默认值：1e-4；
- verbosity：控制优化过程中显示的冗余，设置为 0 即静默。默认值：1。

101：可用的选项是：

- LBGradientStep：梯度差分近似的步长下界。默认值：1e-11；
- MaxIter：最大迭代次数。默认值：15000；
- SpaceDilation：空间膨胀系数。默认值：2.5；
- TolFun：精度参数（关于目标函数）。默认值：1e-6；
- TolX：精度参数（关于工具）。默认值：1e-6；
- verbosity：控制优化过程中显示的冗余，设置为 0 即静默。默认值：1。

102：MATLAB 全局优化工具箱的文档给出了可用的选项。

示例

为了更改 csminwel（mode_compute=4）的默认设置：

```
estimation(...,mode_compute=4,optim=('NumgradAlgorithm',3,'TolFun',1e-5),...);
```

nodiagnostic：不为 Metropolis-Hastings 算法计算收敛诊断。默认值：计算并显示诊断信息。

bayesian_irf：触发 IRFs 后验分布的计算。irf 选项控制 IRFs 的长度。结果存储在 oo_.PostiorIRF.dsge（变量描述参见后文）。

relative_irf：参见 *relative_irf*。

dsge_var=DOUBLE：触发 DSGE-VAR 模型的估计，其中 VAR 模型的 DSGE 先验分布权重校准为传递的值（参见 Del Negro 和 Schorfheide，2004）。它表示虚拟与实际观测的比值。为了保证先验分布的正确，这个值必须大于$(k+n)/T$，其中k是待估参数的数量，n是可观测的数量，T是观测的数量。注意：以前的方法是声明 dsge_prior_weight 为参数，然后校准，现已不支持，将在未来版本的 Dynare 删除。估计过程中产生的一些对象和值存储在 oo_.dsge_var.posterior_mode。

dsge_var：触发 DSGE-VAR 模型的估计，其中 VAR 模型的 DSGE 先验分布权重将被估计（如 Adjemian 等，2008）。DSGE 先验权重的先验分布 dsge_prior_weight，必须在 estimated_params 部分定义。注意：以前的方法声明 dsge_prior_weight 为参数，然后放入 estimated_params，现已不支持，将在未来版本的 Dynare 删除。

dsge_varlag=INTEGER：估计 DSGE-VAR 模型的滞后数。默认值：4。

posterior_sampling_method=NAME：选择抽样选项，以便用于贝叶斯估计过程中从后验分布中得到的样本。默认值：random_walk_metropolis_hastings。

- random_walk_metropolis_hastings：指示 Dynare 使用随机游走 Metropolis-Hastings。算法的提议密度在每一步都被重新输入到之前的抽样中；
- tailored_random_block_metropolis_hastings：指示 Dynare 使用由 Chib 和 Ramamurthy（2010）提出的定制随机游走（Tailored randomized block，TaRB）Metropolis-Hastings 算法，而不是标准的随机游走 Metropolis-Hastings。算法每次迭代时，估计的参数被随机分配到不同的模块。每个模块执行一个寻找众数的步骤。众数中计算的逆海塞矩阵被用作随机游走 Metropolis-Hastings 步骤的提议密度的协方差。如果数值海塞矩阵非正定，则使用 Schnabel 和 Eskow（1990）的广义 Cholesky 分解，但不旋转。TaRB-MH 算法大大降低了 MH 抽样过程中的自相关，从而减少了从后验中抽取代表性样本所需的抽样次数。然而，算法需要更多的时间运行，提高了计算成本；

- independent_metropolis_hastings：使用独立 Metropolis-Hastings 算法，与随机游走 Metropolis-Hastings 算法不同，提议分布不依赖于链的状态；
- slice：指示 Dynare 使用 Planas、Ratto 和 Rossi（2015）的切片采样器。注意，slice 与 prior_trunc=0 不兼容。

posterior_sampler_options=(NAME,VALUE,...)：NAME 和 VALUE 的配对列表，可用于后验抽样方法的选项设置。可用选项的集合取决于所选的后验抽样程序（即关于选项 *posterior_sampling_method* 的值）：random_walk_metropolis_hastings。

可用的选项是：

- proposal_distribution：指定用于提议密度的统计分布；
- rand_multivariate_normal：使用多元正态分布，这是默认值；
- rand_multivariate_student：使用多元学生分布；
- student_degrees_of_freedom：指定多元学生分布的自由度。默认值：3；
- use_mh_covariance_matrix：指示使用前一个运行的 MCMC 抽样的协方差矩阵来定义提议分布的协方差，要求指定 *load_mh_file*。默认值：0；
- scale_file：提供_mh_scale.mat 文件的名称，文件存储了之前运行的 mode_compute=6 调整后的比例系数；
- save_tmp_file：以状态栏的刷新频率将 MCMC 抽样保存到_mh_tmp_blck 文件中，而不是只在当前_mh*_blck 文件存满时，才保存抽样。默认值：0；
- independent_metropolis_hastings：采用与 random_walk_metropolis_hastings 相同的选项；
- slice；
- rotated：使用初始预热迭代的协方差矩阵触发旋转切片迭代。需要 use_mh_covariance_matrix 或 slice_initialize_with_mode。默认值：0；
- mode_files：对于多峰后验，提供一个文件名称，文件包含存储不同的众数，名为 xparams 的 nparam×nmodes 变量。该数组必须具有每个众数的一个列向量和行维度的待估参数。有了这些信息，代码将自动触发 rotated 和 mode 选项。默认值：[]；
- slice_initialize_with_mode：切片的默认值是设置 mode_compute=0，并从先验空间中的一个随机位置开始链。这个选项首先运行寻找众数的程序，然后从众数处启动链。与 rotated 一起，它将使用众数处的逆海塞矩阵执行旋转切片迭代。默认值：0；
- initial_step_size：在 stepping-out 程序中设置初始区间大小，作

为先验支持的一部分。即初始大小为 initial_step_size*(UB-LB)。initial_step_size 必须是区间[0,1]中的实数。默认值：0.8；

- use_mh_covariance_matrix：参见 *use_mh_covariance_matrix*，必须与 rotated 一起使用。默认值：0；
- save_tmp_file：参见 *save_tmp_file*。默认值：1；
- tailored_random_block_metropolis_hastings；
- proposal_distribution：指定用于提议密度的统计分布，参见 *proposal_distribution*；
- new_block_probability=DOUBLE：执行 TaRB Metropolis-Hastings 算法中的随机模块化时，指定属于新模块的下一个参数的概率。数值越高，模块平均大小就越小，每次参数扫描期间形成的随机模块越多。默认值：0.25；
- mode_compute=INTEGER：为 TaRB Metropolis-Hastings 算法的每个模块指定每次迭代中运行的众数搜索程序。参见 mode_compute。默认值：4；
- optim=(NAME,VALUE,...)：指定 TaRB Metropolis-Hastings 算法中使用的众数搜索程序的选项，参见 *optim*；
- scale_file：参见 *scale_file*；
- save_tmp_file：参见 *save_tmp_file*。默认值：1。

moments_varendo：触发内生变量理论矩后验分布的计算。结果存储在 oo_.PosteriorTheoreticalMoments（参见 *oo_.PosteriorTheoreticalMoments*）。ar 选项控制自相关函数的滞后数。

contemporaneous_correlation：参见 *contemporaneous_correlation*，结果存储在 oo_.PosteriorTheoreticalMoments。注意：nocorr 选项没有作用。

no_posterior_kernel_density：关闭后验对象的核密度估计量的计算（参见 *density* 字段）

conditional_variance_decomposition=INTEGER

conditional_variance_decomposition=[INTEGER1:INTEGER2]

conditional_variance_decomposition=[INTEGER1 INTEGER2...]

计算特定时期条件方差分解的后验分布，周期必须严格为正。条件方差由$var(y_{t+k}|t)$给出。对于时期 1，条件方差分解提供了冲击造成的影响效应的分解。结果存储在 oo_.PosteriorTheoreticalMoments.dsge.ConditionalVarianceDecomposition。注意，此选项需要指定选项 moment_varendo。存在测量误差的情况下，该字段将包含去除测量误差后的方差贡献，即分解实际变量而不是测量变量。测量变量的方差分解将存储在 oo_.PosteriorTheoreticalMoments.dsge.ConditionalVarianceDecompositionME。

filtered_vars：触发滤波内生变量/一步向前预测的后验分布的计算，即$E_t y_{t+1}$。结果存储在 oo_.FilteredVariables（描述参见下文）。

smoother：触发平滑内生变量和冲击的后验分布的计算，即给定截止到最终时期（$E_T y_t$）观测变量的所有观测值的可用信息情况下，变量和冲击的期望值。结果存储在 oo_.SmoothedVariables、oo_.SmoothedShocks 和 oo_.SmoothedMeasurementErrors。还会触发 oo_.UpdatedVariables 的计算，其中包含了给定当前日期（$E_t y_t$）可用信息的情况下对变量期望值的估计。这些变量的详细内容见下文。

smoother_redux：大型模型中出发更快的平滑内生变量和冲击的计算。该选项只平滑状态变量（和最大似然估计的表述一样），剩下的变量只进行事后计算。恢复静态未观察到的对象（过滤、平滑、更新、提前k步），但有个完全恢复的例外，取决于所采用的状态空间模型中如何约束静态未观测变量。例如，仅用于恢复未观察到的静态变量的滞后冲击将无法恢复。对于这种例外，仅提供以下输出：

- FilteredVariablesKStepAhead：全部恢复；
- SmoothedVariables,FilteredVariables,UpdatedVariables：在$d+1$期之后的所有期间恢复，其中d表示扩散滤波步骤的数量；
- FilteredVariablesKStepAheadVariances,Variance 和 State_uncertainty：无法恢复，并提供零作为输出。

如果需要这些变量的方差，不要设置选项，或者声明变量为观察到的，使用 NaN 作为数据点。

forecast=INTEGE：计算用于估计的样本结束后在 INTEGE 周期的预测的后验分布。如果没有计算 Metropolis-Hastings，结果存储在变量 oo_.forecast，对应后验众数的预测。如果计算 Metropolis-Hastings，则预测分布存储在变量 oo_.PointForecast 和 oo_.MeanForecast。这些变量的描述参见 4.20 预测。

tex：参见 *tex*。

kalman_algo=INTEGE

0：自动为平稳模型使用多变量卡尔曼滤波，自动为非平稳模型使用多变量扩散卡尔曼滤波；

1：使用多变量卡尔曼滤波；

2：使用单变量卡尔曼滤波；

3：使用多变量扩散卡尔曼滤波；

4：使用单变量扩散卡尔曼滤波。

默认值为 0。当单个或所有序列的观测值缺失时，Dynare 将这些缺失值视为未观测状态，并使用卡尔曼滤波来推断结果（具体信息参见 Durbin 和 Koopman，2012，第 4.10 节）。优点是能够处理某些变量的预测误差方差矩阵变为奇异的观测结果。如果发生这种情况，

相应的观测值在对数似然函数中输入的权重为 0，也就是说，对于每个变量的观测值从似然计算中删除（参见 Durbin 和 Koopman，2012，第 6.4、7.2.5 节以及 Koopman 和 Durbin，2000）。如果指定使用多变量卡尔曼滤波，并且遇到奇异值，Dynare 默认会自动切换到单变量卡尔曼滤波进行参数抽样。这种行为可以通过 *use_univariate_filters_if_singularity_is_detected* 选项来改变。

fast_kalman_filter：根据 Herbst（2015）的描述，使用 Chandrasekhar 递归选择快速卡尔曼滤波。此设置仅用于 kalman_algo=1 或 kalman_algo=3。在使用扩散卡尔曼滤波（kalman_algo=3/lik_init=3）时，可观测数据必须是固定的。这个选项还不兼容 *analytic derivation*。

kalman_tol=DOUBLE：在卡尔曼滤波中确定预测误差的协方差矩阵的奇异性的数值精度（矩阵条件数的最小允许倒数）。默认值是 1e-10。

diffuse_kalman_tol=DOUBLE：在扩散卡尔曼滤波期间，确定预测误差的协方差矩阵奇异性（F_∞）和非平稳状态变量的协方差矩阵的秩（P_∞）的精度。默认值：1e-6。

filter_covariance：保存预测协方差矩阵一步向前误差序列。利用 Metropolis 采样算法，它们储存在 *oo_.FilterCovariance*，否则储存在 *oo_.Smoother.Variance*。如果设置了 filter_step_ahead，还储存预测协方差矩阵的k步向前误差。

filter_step_ahead=[INTEGER1:INTEGER2]

filter_step_ahead=[INTEGER1 INTEGER2...]：触发k步向前滤波值的计算，例如$E_t y_{t+k}$，存储在 oo_.FilteredVariablesKStepAhead。同时，将一步向前值储存在 oo_.FilteredVariables。如果使用 filter_covariance 选项，也会存储 oo_.FilteredVariablesKStepAheadVariances。

filter_decomposition：触发计算上述k步前滤波值的冲击分解。结果存储在 oo_.FilteredVariablesShockDecomposition。

smoothed_state_uncertainty：触发上述k步向前滤波值的方差分解的计算，例如$var_T(y_t)$，结果储存在 oo_.Smoother.State_uncertainty 中。

diffuse_filter：使用扩散卡尔曼滤波（如 Durbin 和 Koopman，2012，；Koopman 和 Durbin（2003）描述的多变量滤波和 Koopman 和 Durbin，2000，描述的单变量滤波）来估计具有非平稳观测变量的模型。当使用 diffuse_filter 时，lik_init 估计选项不起作用。当模型存在非平稳的外生变量时，不存在唯一的确定性稳态。例如，如果生产率是纯粹的随机游走过程：

$$a_t = a_{t-1} + e_t$$

a的任何值$\bar{a}$都是生产率确定性稳态。因此，该模型允许无限个稳态。用户此时必须帮助 Dynare 选择一个稳态，除非零是一个无关紧要的模型稳态，当模型声明使用 linear 选项时才会发生这种情形。如果有一个封闭解存在，用户可以使用 steady_state_model

模块向 Dynare 提供稳态（或者编写一个稳态文件），参见 *steady_state_model*，或者指定一些稳态的约束条件，参见 *equation_tag_for_conditional_steady_state*，以便 Dynare 基于前定条件计算稳态。两种情况下，思路都是用虚拟值表示外生非平稳变量的稳态水平。

注意，模型的非平稳变量必须协整（一阶或k阶差分必须是平稳的）。

heteroskedastic_filter：在 heteroskedastic_shocks 模块使用所提供的异方差的定义滤波、似然估计以及平滑。

selected_variables_only：只为 estimation 命令后列出的变量运行经典平滑。此选项与经典频率学派预测不兼容，在本例中将被拒绝。使用贝叶斯估计时，默认情况只为声明的内生变量运行平滑。默认值：对所有声明的内生变量运行平滑。

cova_compute=INTEGER：等于 0 时，后验众数（或极大似然）计算后，未计算估计参数的协方差矩阵。这提高了开发大型模型的计算速度，而这些信息并非必需。当然，它会打断所有需要协方差矩阵的连续计算。否则，如果这个选项等于 1，则计算协方差矩阵，并将其存储在 MODEL_FILENAME_mode.mat 的变量 hh 中。默认值：1。

solve_algo=INTEGER：参见 *solve_algo*。

order=INTEGER：确定性稳态附近的近似阶数。大于 1 时，使用粒子或非线性滤波评估似然函数（参见 Fernández-Villaverde 和 Rubio-Ramírez ，2005）。默认值：1，使用标准卡尔曼滤波来计算线性化模型的似然函数。

irf=INTEGER：参见 *irf*，仅用于 *bayesian_irf* 之后。

irf_shocks=(VARIABLE_NAME[[,]VARIABLE_NAME...])：参见 *irf_shocks*，仅用于 *bayesian_irf* 之后。

irf_plot_threshold=DOUBLE：参见 *irf_plot_threshold*，仅用于 *bayesian_irf* 之后。

aim_solver：参见 *aim_solver*。

sylvester=OPTION：参见 *sylvester*。

sylvester_fixed_point_tol=DOUBLE：参见 *sylvester_fixed_point_tol*。

lyapunov=OPTION：确定求解 Lyapunov 方程的算法，利用状态变量的稳态初始化卡尔曼滤波的方差－协方差矩阵。可能的选项是：

- default：使用基于 Bartel-Stewart 算法的默认解法求解 Lyapunov 方程；
- fixed_point：采用不动点算法求解 Lyapunov 方程。对于大型模型，这种方法比 default 方法快，但是可能需要大量迭代；
- doubling：采用加倍算法求解 Lyapunov 方程（disclyap_fast）。对于大型模型，这种方法比前两种方法快；
- square_root_solver：使用平方根算法求解 Lyapunov 方程（dlyapchol）。该方法适用于大型模型。（如果安装了控制系统工具箱，可在 MATLAB 环境下使

用；如果安装了 Octave-Forge 的控制包，可在 Octave 环境下使用）。

默认值：`default`。

`lyapunov_fixed_point_tol=DOUBLE`：不动点 Lyapunov 求解程序的收敛准则。默认值：`1e-10`。

`lyapunov_doubling_tol=DOUBLE`：加倍算法求解 Lyapunov 方程的收敛准则。默认值：`1e-16`。

`use_penalized_objective_for_hessian`：用惩罚目标代替目标函数来计算众数处的海塞矩阵。对于某些扰动，当惩罚函数降低后验密度（或似然）的值，Dynare 无法求解模型（稳态存在问题、BK 条件……）。在实践中，只有当发现后验众数靠近模型行为不良表现的区域时，惩罚目标和原始目标才会不同。默认值：使用原始目标函数。

`analytic_derivation`：`order=1` 处触发解析梯度的估计。最终的海塞矩阵也进行了解析计算。仅用于平稳模型却无缺失观测值的情形，例如 `kalman_algo<3`。优化算子取决于解析梯度的设定 `mode_compute=1`、`3`、`4`、`5`、`101`。

`ar=INTEGER`：参见 *`ar`*，只与选项 `moments_varendo` 一起使用。

`endogenous_prior`：使用 Christiano、Trabandt 和 Walentin（2011）的内生先验分布设置。该过程是受序贯贝叶斯学习的启发。从 `estimated_params` 模块中指定参数的独立初始先验分布开始，在“预样本”中观察到的标准偏差（被视为实际样本）用于更新初始先验分布。因此，初始先验和可观测变量标准偏差的预样本似然函数的乘积用作新的先验分布（更多信息参见 Christiano，Trabandt 和 Walentin，2011，的技术附录）。这种程序在常规后验估计导致模型变量方差过大时特别有帮助，常规后验估计程序会最小化样本内预测误差（统计量没有明确的目标，但通常研究人员对这种方法特别感兴趣）。

`use_univariate_filters_if_singularity_is_detected=INTEGER`：如果在似然计算中遇到奇异问题，该选项决定 Dynare 是否应自动切换到单变量滤波（如果选项等于 `1`，则为自动切换）。或者如果等于 `0`，Dynare 不会自动切换滤波，而是在遇到此类奇异问题时使用惩罚值。默认值：`1`。

`keep_kalman_algo_if_singularity_is_detected`：使用默认 *`use_univariate_filters_if_singularity_is_detected=1`* 时，Dynare 在卡尔曼滤波过程中遇到奇异预测误差方差矩阵时，会切换到单变量卡尔曼滤波。首次遇到这种奇异问题时，所有后续参数抽样和计算将自动依赖于单变量滤波，即 Dynare 不会尝试多元滤波。使用 `keep_kalman_algo_if_singularity_is_detected` 选项来约束 `use_univariate_filters_if_singularity_is_detected` 选项仅影响当前抽样/计算的行为。

`rescale_prediction_error_covariance`：重新调节卡尔曼滤波的预测误差协方差，以避免不恰当缩放的矩阵，降低切换到单变量卡尔曼滤波器的概率（后者较慢）。默认值：不进行重新调节。

qz_zero_threshold=DOUBLE：参见 *qz_zero_threshold*。

taper_steps=[INTEGER1 INTEGER2...]：Geweke（1992，1999）收敛诊断中用于光谱窗口的缩减百分比（要求 *mh_nblocks=1*）。缩减用于考虑后验抽取的序列相关性。默认值：[4 8 15]。

geweke_interval=[DOUBLE DOUBLE]：舍弃第一个 mh_drop=DOUBLE 百分比的抽取为预热迭代之后，在 MCMC 链首末位置计算 Geweke（1992，1999）收敛诊断（需要 mh_nblocks=1）的 MCMC 抽样百分比。默认值：[0.2 0.5]。

raftery_lewis_diagnostics：触发 Raftery 和 Lewis（1992）收敛诊断计算。目标是提供要求的抽样数估计一个概率 s 且精度 r 的累积分布函数的特定 q 分位数。通常，人们希望估计出 95%的概率下（s=0.95），精度为 0.5%（r=0.005）的 q=0.025 分位数（对应于 95%的 HPDI）。默认值可以通过 *raftery_lewis_qrs* 改变。根据一阶马尔科夫链理论，诊断将提供所需的预热（M）、预热之后的抽样数（N），以及传递一阶链的稀释因子（k）。表的末行还提供所有参数的最大值。

raftery_lewis_qrs=[DOUBLE DOUBLE DOUBLE]：在 Raftery 和 Lewis（1992）收敛诊断中，设置概率为 s，精度为 r 的累积分布函数的 q 分位数。默认值：[0.025 0.005 0.95]。

consider_all_endogenous：计算所有内生变量的后验矩、平滑变量、*k*步向前滤波变量和预测（当要求时），等价于手动在 estimation 命令后列出所有内生变量。

consider_all_endogenous_and_auxiliary：计算所有内生变量和预处理器引入的辅助变量后验矩、平滑变量、*k*步向前过滤变量和预测（当要求时）。例如，在卡尔曼平滑器的结果上运行 smoother2histval 时此选项很有用。

consider_only_observed：计算所有观察变量的后验矩、平滑变量、*k*步向前滤波变量和预测（当要求时），等价于手动在 estimation 命令后列出所有观测变量。

number_of_particles=INTEGE：计算非线性状态空间模型的似然函数时使用的粒子数。默认值：1000。

resampling=OPTION：确定是否对粒子进行重复采样。可能的选项是：

- none：不重复抽样；
- systematic：在每次迭代时重新抽样，这是默认值；
- generic：当且仅当有效样本量低于 resampling_threshold**number_of_particle* 定义的特定水平时重采样。

resampling_threshold=DOUBLE：介于 0 到 1 之间的实数。当有效粒子数小于该选项的数值×粒子总数时，就会触发重采样步骤（正如 number_of_particles）。当且仅当选项 *resampling* 使用 generic 时有效。

resampling_method=OPTION：设置重抽样方法，可能的选项是：kitagawa、st

ratified 和 smooth。

filter_algorithm=OPTION：设置粒子滤波算法。可能的选项是：

- sis：顺序重要性抽样算法，这是默认值；
- spf：辅助粒子滤波；
- gf：高斯滤波；
- gmf：高斯混合滤波；
- cpf：条件粒子滤波；
- nlkf：采用非线性测量和状态方程的标准（线性）卡尔曼滤波算法。

proposal_approximation=OPTION：设置提议分布的近似方法。可能的选项是：cubature、montecarlo 和 unscented。默认值：unscented。

distribution_approximation=OPTION：设置近似粒子分布的方法。选项的可能值包括：cubature、montecarlo 和 unscented。默认值：unscented。

cpf_weights=OPTION：控制用于更新条件粒子滤波中权重的方法，可能的值为 amisanotristani（Amisano 等，2010）或者 murrayjonesparslow（Murray 等，2013）。默认值：amisanotristani。

nonlinear_filter_initialization=INTEGER：设置非线性滤波的初始条件。默认情况下，非线性滤波用状态变量的无条件协方差矩阵初始化，由模型的一阶近似的简化式计算得出。如果 nonlinear_filter_initialization=2，用模型二阶近似的简化式解的随机模拟的估计协方差矩阵初始化非线性滤波。这两种初始化都假定模型是平稳的，如果模型有单位根（估计之前可以使用 *check*）则不能使用。如果模型有随机趋势，用户必须使用 nonlinear_filter_initialization=3，使用状态变量的协方差矩阵的单位矩阵初始化滤波。默认值：nonlinear_filter_initialization=1（基于模型的一阶近似初始化）。

particle_filter_options=(NAME,VALUE,...)：一系列 NAME 和 VALUE 的配对，为粒子滤波程序设置一些细粒度的选项，可用的选项集取决于所选的过滤程序。更多粒子滤波的设定信息参见 https://git.dynare.org/Dynare/dynare/-/wikis/Particle-filters。

可用的选项有：

- pruning：剪枝粒子滤波模拟。默认值：不进行；
- liu_west_delta：Liu/West 在线滤波的 delta 值。默认值：0.99；
- unscented_alpha：无迹转换的 alpha 值。默认值：1；
- unscented_beta：无迹转换的 beta 值。默认值：2；
- unscented_kappa：无迹转换的 kappa 值。默认值：1；
- initial_state_prior_std：nonlinear_filter_initialization=3 时，选择初始状态变量的协方差矩阵对角线上的值。默认值：1；

- mixture_state_variables：状态变量的高斯混合滤波混合成分数量。默认值：5；
- mixture_structural_shocks：结构性冲击的高斯混合滤波混合成分数量。默认值：1；
- mixture_measurement_shocks：测量误差的高斯混合滤波混合成分数量。默认值：1。

注意事项

如果 estimated_params 的参数没有使用 mh_jscale 参数，该过程对所有参数使用 mh_jscale。如果未设置 mh_jscale 选项，则该过程对所有参数使用 0.2。注意，如果使用 mode_compute=6 或指定调用 scale_file 的 posterior_sampler_option，则 estimated_params 中设置的值将被覆盖。

“内生”先验约束

在估计过程中，还可以对模型的 IRFs 和矩施加隐性“内生”先验。例如，可以指定模型的所有有效参数抽样必须生成大于 1 的财政乘数，指定政府支出冲击的脉冲响应必须产生多大的影响。先验约束可以通过 irf_calibration 和 moment_calibration 模块（参见 4.22.2 IRF/矩校准）施加。内部工作原理是，所有与上述模块中提供的“校准值”不一致的参数抽样都被舍弃，即指定先验密度为 0。指定这些模块时，重要的是此时不能很容易地实施 model_comparison，因为先验密度之和不等于 1。

输出

运行估计后，冲击的参数 M_.params 和方差矩阵 M_.Sigma_e 设置为不用 Metropolis 采样算法迭代计算的极大似然估计或后验众数。运行了 Metropolis 采样算法迭代的估计（选项 mh_replic>0 或者选项 load_mh_file）后，冲击的参数 M_.params 和方差矩阵 M_.Sigma_e 设为后验均值。

根据选项，estimation 将结果存储在 oo_结构的不同字段，如下所述。以下变量将对特定字段名采用缩写方式：

MOMENT_NAME：此字段可以采用以下值：

- HPDinf：90%HPD 区间的下界④；
- HPDsup：90%HPD 区间的上界；
- HPDinf_ME：考虑测量误差时观测值 90%HPD 区间⑤的下限（参见 Christoffel 等，2010，第 17 页）；
- HPDsup_ME：考虑测量误差时观测值 90%HPD 区间的上限；
- Mean：后验分布均值；
- Median：后验分布中位数；

④ 参见选项 *conf_sig* 来改变 HPD 区间的大小。
⑤ 参见选项 *conf_sig* 来改变 HPD 区间的大小。

- Std：后验分布标准差；
- Variance：后验分布方差；
- decile：后验分布的十分位数；
- density：后验密度的非参数估计值，采用 Skoeld 和 Roberts（2003）概述的方法。第一列和第二列分别为横坐标和坐标。

ESTIMATED_OBJECT：此字段可以采用以下值：

- measurement_errors_corr：两个测量误差的相关系数；
- measurement_errors_std：测量误差标准差；
- parameters：参数；
- shocks_corr：两个结构冲击之间的相关系数；
- shocks_std：结构冲击的标准差。

MATLAB/Octave variable:oo_.MarginalDensity.LaplaceApproximation

变量是由 estimation 命令设置，存储了基于拉普拉斯近似的边际数据密度。

MATLAB/Octave variable:oo_.MarginalDensity.ModifiedHarmonicMean

如果与 mh_replic>0 或 load_mh_file 选项一起使用，变量是由 estimation 命令设置。存储基于 Geweke（1999）改进的调和平均估计量（Modified Harmonic Mean estimator）的边际数据密度。

MATLAB/Octave variable:oo_.posterior.optimization

如果使用众数搜索选项，变量是由 estimation 命令设置。在众数处存储结果。字段形式：

```
oo_.posterior.optimization.OBJECT
```

其中 OBJECT 是以下内容之一：

- mode：众数的参数向量；
- Variance：众数的逆海塞矩阵或与选项 MCMC_jumping_covariance 一起使用时 MCMC 跳跃协方差矩阵；
- log_density：当使用 mode_compute>0 时，存储在众数的对数似然比（ML）/对数后验密度（Bayesian）。

MATLAB/Octave variable:oo_.posterior.metropolis

如果使用 mh_replic>0，变量是由 estimation 命令设置。字段形式：

```
oo_.posterior.metropolis.OBJECT
```

其中 OBJECT 是以下内容之一：

- mean：来自 MCMC 的参数平均值向量；
- Variance：MCMC 中参数抽样的协方差矩阵。

MATLAB/Octave variable:oo_.FilteredVariables

如果与 filter_vars 选项一起使用，变量是由 estimation 命令设置。

无 Metropolis 采样算法的估计之后，字段形式：

```
oo_.FilteredVariables.VARIABLE_NAME
```

有 Metropolis 采样算法的估计之后，字段形式：

```
oo_.FilteredVariables.MOMENT_NAME.VARIABLE_NAME
```

MATLAB/Octave variable:oo_.FilteredVariablesKStepAhead

如果与 filter_step_ahead 选项一起使用，变量是由 estimation 命令设置。k步存储在行中，而列表示各自的变量，第三维度提供了预测的观测值结果。例如，如果 filter_step_ahead=[1 2 4]和 nobs=200，元素（3，5，204）存储变量 5 在时间$t=200$超前四个周期的滤波值，用于时间$t=204$。样本的开始和结束时不能预测的时期，即示例中的条目（1，5，1）和（1，5，204）设置为零。注意，贝叶斯估计的情况下，变量在 estimation 命令之后按声明顺序排列（如果没有指定变量，则按一般的声明顺序排列）。在运行经典平滑器的情况下，变量将始终按一般声明顺序排列。如果选项是 *selected_variables_only*，非指定变量将被简单地排除在这个顺序。

MATLAB/Octave variable:oo_.FilteredVariablesKStepAheadVariances

如果与 filter_step_ahead 选项一起使用，变量是由 estimation 命令设置。它是一个四维数组，k步存储于第一维度，而第四维度提供了预测的观测值结果。第二维度和第三维度提供各自的变量。例如，如果 filter_step_ahead=[1 2 4]和 nobs=200，元素（3，4，5，204）存储变量 4 和变量 5 之间在时间$t=200$超前四个周期的预测误差协方差，用于时间$t=204$。填充零和变量的排序类似于 oo_.FilteredVariablesKStepAhead。

MATLAB/Octave variable:oo_.Filtered_Variables_X_step_ahead

贝叶斯估计情形下，如果与 filter_step_ahead 选项一起使用，变量是由 estimation 命令设置。字段形式：

```
oo_.Filtered_Variables_X_step_ahead.VARIABLE_NAME
```

第n个条目存储在时间n的k步超前滤波变量，用于时间$n+k$。

MATLAB/Octave variable:oo_.FilteredVariablesShockDecomposition

如果与 filter_step_ahead 选项一起使用，变量是由 estimation 命令设置。k步存储在行中，而列表示各自的变量，第三个维度对应声明顺序的冲击，而第四维度提供了预测的观测值结果。例如，如果 filter_step_ahead=[1 2 4]和 nobs=200，元素（3，5，2，204）存储对变量 5（与平均值的偏差）第二次冲击在时间$t=200$超前四个周期滤波的贡献，用于时间$t=204$。样本的开始和结束时不能预测的时期，即示例中的条目（1，5，1）和（1，5，204）设置为零。填充零和变量的排序类似于 oo_.FilteredVariablesKStepAhead。

MATLAB/Octave variable:oo_.PosteriorIRF.dsge

如果与 bayesian_irf 选项一起使用，变量是由 estimation 命令设置。字段形式：

```
oo_.PosteriorIRF.dsge.MOMENT_NAME.VARIABLE_NAME_SHOCK_NAME
```

MATLAB/Octave variable:oo_.SmoothedMeasurementErrors

如果与 smoother 选项一起使用，变量是由 estimation 命令设置。字段形式：

```
oo_.SmoothedMeasurementErrors.VARIABLE_NAME
```

MATLAB/Octave variable:oo_.SmoothedShocks

变量是由 estimation 命令设置（如果与 smoother 选项一起使用），或者由 calib_smoother 命令设置。

无 Metropolis 采样算法估计之后，或者 calib_smoother 计算之后，字段形式：

```
oo_.SmoothedShocks.VARIABLE_NAME
```

有 Metropolis 采样算法估计之后，字段形式如下：

```
oo_.SmoothedShocks.MOMENT_NAME.VARIABLE_NAME
```

MATLAB/Octave variable:oo_.SmoothedVariables

变量是由 estimation 命令设置（如果与 smoother 选项一起使用），或者由 calib_smoother 命令设置。

无 Metropolis 采样算法估计之后，或者 calib_smoother 计算之后，字段形式：

```
oo_.SmoothedVariables.VARIABLE_NAME
```

有 Metropolis 采样算法估计之后，字段形式：

```
oo_.SmoothedVariables.MOMENT_NAME.VARIABLE_NAME
```

MATLAB/Octave command:get_smooth('VARIABLE_NAME'[,'VARIABLE_NAME']...);

返回给定内生或外生变量的平滑值，因为它们存储在 oo_.SmoothedVariables 和 oo_.SmoothedShocks 变量中。

MATLAB/Octave variable:oo_.UpdatedVariables

变量是由 estimation 命令设置（如果与 smoother 选项一起使用），或者由 calib_smoother 命令设置。包含给定当前日期可用信息的变量期望值的估计。

无 Metropolis 采样算法估计之后，或者 calib_smoother 计算之后，字段形式：

```
oo_.UpdatedVariables.VARIABLE_NAME
```

有 Metropolis 采样算法估计之后，字段形式：

```
oo_.UpdatedVariables.MOMENT_NAME.VARIABLE_NAME
```

MATLAB/Octave command:get_update('VARIABLE_NAME'[,'VARIABLE_NAME']...);

返回给定变量的更新值，因为它们存储在 oo_.UpdatedVariables 变量中。

MATLAB/Octave variable:oo_.FilterCovariance

如果已请求 filter_covariance 选项，并与 smoother 和 Metropolis 采样算法一起使用，则由 estimation 命令设置三维数组。包含一系列来自卡尔曼平滑的超前预测误差协方差矩阵。M_.endo_nbr×M_.endo_nbr×（T+1）数组包含沿前两个维度按声明顺序排列的变量。数组的第三维提供了已对其进行预测的观测值。字段形式：

```
oo_.FilterCovariance.MOMENT_NAME
```

注意，不支持密度估计。

MATLAB/Octave variable:oo_.Smoother.Variance

如果已请求 filter_covariance 选项，没有 Metropolis 采样算法的 estimation 命令（如果与 smoother 选项一起使用）或 calib_smoother 命令设置的三维数组。包含一系列来自卡尔曼平滑的超前预测误差协方差矩阵。M_.endo_nbr×M_.endo_nbr×（T+1）数组包含沿前两个维度按声明顺序排列的变量，第三个维度提供了已预测的观测值。

MATLAB/Octave variable:oo_.Smoother.State_uncertainty

如果已经请求了 smoothed_state_uncertainty 选项，没有 Metropolis 采样算法的 estimation 命令（如果与 smoother 选项一起使用）或者 calib_smoother 命令设置的三维数组。包含一系列来自卡尔曼平滑的完整数据的状态估计的协方差矩阵。M_.endo_nbr×M_.endo_nbr×T 数组包含沿前两个维度按声明顺序排列的变量，第三个维度提供了平滑估计的观测值。

MATLAB/Octave variable:oo_.Smoother.SteadyState

没有 Metropolis 采样算法的 estimation 命令设置（如果与 smoother 选项一起使用），或者 calib_smoother 命令设置的变量。包含按变量声明顺序排序在平滑器中使用的内生变量稳态分量。

MATLAB/Octave variable:oo_.Smoother.TrendCoeffs

没有 Metropolis 采样算法的 estimate 命令（如果与 smoother 选项一起使用）设置或者由 calib_smooth 命令设置的变量。包含按观察变量声明顺序排序在平滑器中使用的观察变量趋势系数。

MATLAB/Octave variable:oo_.Smoother.Trend

变量是由 estimation 命令设置（如果与 smoother 选项一起使用），或者由 calib_smoother 命令设置。包含平滑器中使用的变量趋势组件。字段形式：

```
oo_.Smoother.Trend.VARIABLE_NAME
```

MATLAB/Octave variable:oo_.Smoother.Constant

变量是由 estimation 命令设置（如果与 smoother 选项一起使用），或者由 calib_smoother 命令设置。包含在平滑器中使用的内生变量的常量部分，如使用预滤器选项时的数据平均值。字段形式：

```
oo_.Smoother.Constant.VARIABLE_NAME
```

MATLAB/Octave variable:oo_.Smoother.loglinear

跟踪平滑器是否使用 *loglinear* 选项运行的指示器，因此存储的平滑对象是否在日志中。

MATLAB/Octave variable:oo_.PosteriorTheoreticalMoments

如果和 moments_varendo 选项一起使用，变量由 estimate 命令设置。字段形式：

```
oo_.PosteriorTheoreticalMoments.dsge.THEORETICAL_MOMENT.ESTIMATED_OBJECT.MOMENT_NAME.VARIABLE_NAME
```

其中 *THEORETICAL_MOMENT* 为下列之一：

- covariance：内生变量的方差－协方差；
- contemporaneous_correlation：当指定 *contemporaneous_correlation* 选项时，内生变量的同期相关性；
- correlation：内生变量的自相关和相关。字段是相关度为 1 的向量，最高阶为 options_.ar。；
- VarianceDecomposition：方差的分解（无条件方差，即在水平无穷远处）[6]；
- VarianceDecompositionME：与 VarianceDecomposition 相同，但包含测量变量而非实际变量的分解。测量误差的共同贡献将保存在名为 ME 的字段中；
- ConditionalVarianceDecomposition：仅当已指定 conditional_variance_decomposition 选项时。在存在测量误差的情况下，该字段将包含去除测量误差后的方差贡献，即对实际变量而不是测量变量进行分解；
- ConditionalVarianceDecompositionME：仅当已指定 conditional_variance_decomposition 选项时。与 *ConditionalVarianceDecomposition* 相同，但包含测量变量而非实际变量的分解。测量误差的共同贡献将保存在名称为 ME 的字段中。

MATLAB/Octave variable:oo_.posterior_density

如果与 mh_replica>0 或 load_mh_file 选项一起使用，变量由 estimate 命令设置。字段格式：

```
oo_.posterior_density.PARAMETER_NAME
```

MATLAB/Octave variable:oo_.posterior_hpdinf

如果与 mh_replica>0 或 load_mh_file 选项一起使用，变量由 estimate 命令设置。字段格式：

```
oo_.posterior_hpdinf.ESTIMATED_OBJECT.VARIABLE_NAME
```

⑥ 当冲击相关时，就是按照冲击声明的顺序通过 Cholesky 方法分解正交化冲击（见 4.2 变量声明）。

MATLAB/Octave variable:oo_.posterior_hpdsup

如果与 mh_replica>0 或 load_mh_file 选项一起使用，变量由 estimate 命令设置。字段格式：

```
oo_.posterior_hpdsup.ESTIMATED_OBJECT.VARIABLE_NAME
```

MATLAB/Octave variable:oo_.posterior_mean

如果与 mh_replica>0 或 load_mh_file 选项一起使用，变量由 estimate 命令设置。字段格式：

```
oo_.posterior_mean.ESTIMATED_OBJECT.VARIABLE_NAME
```

MATLAB/Octave variable:oo_.posterior_mode

众数搜索期间 estimate 命令设置的变量。字段格式为：

```
oo_.posterior_mode.ESTIMATED_OBJECT.VARIABLE_NAME
```

MATLAB/Octave variable:oo_.posterior_std_at_mode

众数搜索期间 estimate 命令设置的变量。基于 oo_.posterior_mode 处的逆海塞矩阵。字段格式：

```
oo_.posterior_std_at_mode.ESTIMATED_OBJECT.VARIABLE_NAME
```

MATLAB/Octave variable:oo_.posterior_std

如果与 mh_replica>0 或 load_mh_file 选项一起使用，变量由 estimate 命令设置。字段格式：

```
oo_.posterior_std.ESTIMATED_OBJECT.VARIABLE_NAME
```

MATLAB/Octave variable:oo_.posterior_var

如果与 mh_replica>0 或 load_mh_file 选项一起使用，变量由 estimate 命令设置。字段格式：

```
oo_.posterior_var.ESTIMATED_OBJECT.VARIABLE_NAME
```

MATLAB/Octave variable:oo_.posterior_median

如果与 mh_replica>0 或 load_mh_file 选项一起使用，变量由 estimate 命令设置。字段格式：

```
oo_.posterior_median.ESTIMATED_OBJECT.VARIABLE_NAME
```

示例

以下是一些生成变量的示例：

```
oo_.posterior_mode.parameters.alp
oo_.posterior_mean.shocks_std.ex
oo_.posterior_hpdsup.measurement_errors_corr.gdp_conso
```

MATLAB/Octave variable:oo_.dsge_var.posterior_mode

mode_compute 之后 estimate 命令的 dsge_var 选项设置的结构，保存以下字段：

- PHI_tilde：众数（Del Negro 和 Schorfheide，2004，方程 28）的堆叠后验 DSGE-BVAR 自回归矩阵；
- SIGMA_u_tilde：众数（Del Negro 和 Schorfheide，2004，方程 29）的后验协方差矩阵；
- iXX：众数（$inv(\lambda T\Gamma_{XX}^{*}+X'X)$）的 DSGE-BVAR 中的后验种群矩；
- prior：存储 DSGE-BVAR 先验的结构；
- PHI_star：众数（Del Negro 和 Schorfheide，2004，方程 22）的堆叠先验 DSGE-BVAR 自回归矩阵；
- SIGMA_star：众数（Del Negro 和 Schorfheide，2004，方程 23）下的 DSGE-BVAR 的先验协方差矩阵；
- ArtificialSampleSize：人工先验样本的大小（$inv(\lambda T)$）；
- DF：先验自由度（$inv(\lambda T-k-n)$）；
- iGXX_star：X 之间的理论先验“协方差”的逆（Del Negro 和 Schorfheide 的Γ_{XX}^{*}，2004）

MATLAB/Octave variable:oo_.RecursiveForecast

如果与 nobs=[INTEGER1:INTEGER2]选项一起使用时，变量由 estimate 命令的 forecast 选项设置（参见 *nobs*）。字段格式为：

```
oo_.RecursiveForecast.FORECAST_OBJECT.VARIABLE_NAME
```

其中 FORECAST_OBJECT 是以下之一[⑦]：

- Mean：后验预测分布的平均值；
- HPDinf/HPDsup：仅考虑参数不确定性的 90%HPD 区间的上限/下限（对应 *oo_.MeanForecast*）；
- HPDTotalinf/HPDTotalsup.：考虑参数和未来冲击不确定性的 90% HPD 区间的上限/下限（对应 *oo_.PointForecast*）；
- VARIABLE_NAME：包含以下大小的矩阵，使用 nobs=[INTEGER1:INTEGER2]选项请求预测的时间段数×预测选项请求的预测范围数。即行表示执行预测的时间段，列表示相应的k步超前预测。起始期按升序排列，而不是按声明顺序排列。

MATLAB/Octave variable:oo_.convergence.geweke

当与 mh_nblocks=1 选项一起使用时，变量由 estimation 命令的收敛诊断设置（参见 *mh_nblocks*）。字段格式为：

```
oo_.convergence.geweke.VARIABLE_NAME.DIAGNOSTIC_OBJECT
```

其中 *DIAGNOSTIC_OBJECT* 是以下之一：

⑦ 更多信息参见 *forecast*。

- `Posteriormean`：后验参数分布的平均值；
- `Posteriorstd`：后验参数分布的标准偏差；
- `nse_iid`：独立同分布（i.i.d）假设下抽取的数值标准误差（NSE）；
- `rne_iid`：独立同分布（i.i.d）假设下抽取的相对数值效率（RNE）；
- `nse_x`：使用 x%锥度时的数值标准误差（NSE）；
- `rne_x`：使用 x%锥度时的相对数值效率（RNE）；
- `pooled_mean`：汇集 *`geweke_interval`* 中指定链的首末部分并以相对精度对它们加权时的参数平均值。它是一个向量，包含独立同分布（i.i.d）假设下的结果，后跟使用 `taper_steps` 选项的结果（参见 *`taper_steps`*）；
- `pooled_nse`：汇集链的首末部分并以相对精度对它们加权时参数的 NSE。参见 *`pooled_mean`*；
- `prob_chi2_test`：MCMC 链首尾平均值相等的卡方检验的 p 值。参见 *`pooled_mean`*。高于 0.05 的值表示在 5%的水平上不能拒绝均值相等的原假设，因此不能拒绝收敛性。沿 `taper_steps` 的不同值表示抽取中存在显着的自相关。此时使用更高锥度的估计通常更可靠。

`Command:unit_root_vars` `VARIABLE_NAME...;`

现已废弃。使用 `estimation` 选项 `diffuse_filter` 估计具有非平稳观察变量的模型，或使用 `steady` 选项 `nocheck` 防止 `steady` 检查由稳态文件返回的稳态。

Dynare 还具有估计贝叶斯 VARs 的能力：

`Command:bvar_density;`

使用 Minnesota 先验计算一个估计的 BVAR 模型的边际密度。获取更多信息，参见与 Dynare 发行版一起提供的 `bvar-a-la-sims.pdf`。

4.16 矩估计方法

只要你有对一些内生变量的观察，就有可能使用 Dynare 的矩法估计部分或所有参数。模拟矩估计法（SMM）和广义矩估计法（GMM）都可用。总体思路是最小化无条件模型矩和相应数据矩（所谓正交性或矩条件）之间的距离。对于 SMM，Dynare 通过基于任意阶数扰动近似的随机模拟计算模型矩，而对于 GMM，矩是基于三阶扰动解的修剪状态空间表示以封闭形式计算的。以闭合形式计算模型矩。SMM 的实施灵感来自 Born and Pfeifer（2014）和 Ruge Murcia（2012），而 GMM 的实施灵感来自 Andreasen、Fernández Villaverde 和 Rubio Ramírez（2018）以及 Mutschler（2018）。成功的估计在很大程度上依赖于扰动近似的准确性和效率，因此建议尽可能地调整（参见 4.13.1 计算随机解）。给定一定的正则性条件，矩估计的方法是一致和渐近正态分布的（参见 Duffie 和 Singleton，1993，对 SMM 和汉森，1982，对 GMM）。例如，要求具有至少与估计参数（过度识别或恰好识别）相同的矩条件。此外，估计参数的矩的雅可比行列式要满秩。4.22.3 执行识别分析有助于

检查这种规律性条件。

在过度识别的情况下，声明的矩条件比待估参数多，*weighting_matrix* 的选择对估计的效率很重要，因为估计的正交性条件是不等方差的随机变量，通常是非零交叉矩协方差。加权矩阵允许重新加权力矩，以更加强调信息量更大或测量效果更好的矩条件（从获得更小方差的意义上讲）。为了获得渐近效率，需要选择加权矩阵，以便在适当缩放后，概率极限与正交条件向量的极限分布的协方差矩阵的逆成正比。Dynare 使用带有 Bartlett 核的 Newey-West 型估计器计算所谓的最优加权矩阵的估计。注意，过度识别的情况下，建议至少在两个阶段估计，例如设置 *weighting_matrix=['DIAGONAL','DIAGONAL']*，以便最佳权重矩阵的计算受益于先前阶段的一致估计。最优加权矩阵用于计算标准误差和过度识别约束的 J 检验，检验模型和矩条件的选择是否充分符合数据。如果一个“有效”模型的零假设被拒绝，那么模型或正交条件的选择（很可能）有问题。

如果矩距离函数的（假定的）全局最小值位于通常认为不太可能的参数空间区域（荒谬参数的困境，*dilemma of absurd parameters*），可以选择 *penalized_estimator* 选项。类似于向似然中添加先验信息，该选项将先验信息（即先验平均值）作为附加的矩约束，并根据先验精度加权，以便最小化算法诱导到参数空间中更合理的区域。理想情况下，这些区域的特征是目标函数的值稍差。注意，添加先验信息的代价是损失估计效率。

Command:varobs VARIABLE_NAME...;

要求：*matched_moments* 模块中使用的所有变量都是可观测的。更多细节性信息参见 *varobs*。

Block:matched_moments;

指定用于估算的积矩。目前，只支持线性积矩（例如：$E[y_t]$、$E[y_t^2]$、$E[x_t y_t]$、$E[y_t y_{t-1}]$、$E[y_t^3 x_{t-4}^2]$）。对于$E[\log(y_t)\, e^{x_t}]$等其他函数，需要声明辅助内生变量。模块内的每行应采用以下形式：

```
VARIABLE_NAME(LEAD/LAG)^POWER*VARIABLE_NAME(LEAD/LAG)^POWER
*...*VARIABLE_NAME(LEAD/LAG)^POWER;
```

其中，*VARIABLE_NAME* 是声明的可观察变量的名称，*LEAD/LAG* 是滞后的负整数或超前的正整数，*POWER* 是一个表示变量指数的正整数。可以忽略等于 0 的 *LEAD/LAG* 或等于 1 的 *POWER*。

示例

对于$E[c_t]$、$E[y_t]$、$E[c_t^2]$、$E[c_t y_t]$、$E[y_t^2]$、$E[c_t c_{t+3}]$、$E[y_{t+1}^2 c_{t-4}^3]$、$\mathrm{E}[c_{t-5}^3 y_t^2]$，使用以下模块：

```
matched_moments;
c;
```

```
y;
c*c;
c*y;
y^2;
c*c(3);
y(1)^2*c(-4)^3;
c(-5)^3*y(0)^2;
end;
```

局限性

1．对于 GMM，Dynare 只能计算理论平均值、协方差和自相关系数（即一阶矩和二阶矩）。仅 SMM 支持高阶矩；

2．默认情况下，积矩不会去除均值，除非 prefilter 选项设置为 1。也就是说，默认情况下，$c*c$对应$E[c_t^2]$，而不是$Var[c_t]=E[c_t^2]-E[c_t]^2$。

输出

下列情况，Dynare 将 *matched_moments* 转换为单元数组 M_.matched_moments：

- 第一列按声明顺序包含所选变量的索引向量；
- 第二列包含相应的超前和滞后向量；
- 第三列包含相应的幂向量。

估计阶段，Dynare 将消除 M_.matched_moments 的所有冗余或重复正交条件，并显示哪些条件已被移除。在上面的例子中，最后一行就是这种情况，它与倒数第二行相同。原始模块保存在 M_.matched_moments_orig 中。

Block:estimated_params;

有关含义和语法参见 *estimated_params*。

Block:estimated_params_init;

有关含义和语法参见 *estimated_params_init*。

Block:estimated_params_bounds;

有关含义和语法参见 *estimated_params_bounds*。

Command:method_of_moments(OPTIONS...);

运行矩估计方法。命令窗口中将显示以下信息：

- 用户选择的选项概述；
- 每个阶段和迭代的估计结果；
- 最小矩距目标函数的值；
- J 检验的结果；
- 数据矩和估计模型矩表。

必要选项

mom_method=SMM|GMM：“模拟矩估计法”由 SMM 触发，“广义矩估计法”由 GMM 触发。

datafile=FILENAME：包含数据的文件名称。有关含义和语法参见 *datafile* 的语法和含义。

SMM 和 GMM 常见选项

order=INTEGER：扰动近似的阶数。对于 GMM，仅支持 1/2/3 阶扰动。对于 SMM，可以选择任意阶数。注意，在其他函数中设置的阶数不会覆盖默认值。默认值：1。

pruning：在迭代计算解的模拟时，舍弃高阶项。更多详细信息参见 *pruning*。默认值：不设置为 SMM，总是设置为 GMM。

penalized_estimator：该选项包括估计参数与先前平均值的偏差，作为附加矩约束，并根据先验精度加权。默认值：不设置。

weighting_matrix=['WM1','WM2',...,'WMn']：确定每个估算阶段使用的权重矩阵。元素数量将定义阶段数量（例如：weighting_matrix=['DIAGONAL','DIAGONAL','OPTIMAL']执行三阶段估计）。WM 的可能值为：

- IDENTITY_MATRIX：将权重矩阵设置为等于单位矩阵；
- OPTIMAL：使用带有 Bartlett 核的 Newey-West 型估计计算的最优加权矩阵。第一阶段，数据矩被用作模型矩的初始估计，而在随后的阶段，计算最优加权矩阵时使用模型矩的先前估计；
- DIAGONAL：使用最优权重矩阵的对角线。这种选择将权重放在指定的矩上，而不是它们的线性组合；
- FILENAME：包含用户指定的权重矩阵 mat 文件的名称（扩展名*.mat）。该文件必须包含一个称为*权重矩阵*的正定方阵，两个维度都等于正交条件的数量。

默认值：['DIAGONAL','OPTIMAL']。

weighting_matrix_scaling_factor=DOUBLE：目标函数中权重矩阵的缩放。应选择此值获得合理数值范围内的目标函数值，以防止上溢和下溢。默认值：1。

bartlett_kernel_lag=INTEGER：计算最优加权矩阵的核带宽。默认值：20。

se_tolx=DOUBLE：双边有限差分法计算标准误差时数值微分步长。默认值：1e-5。

verbose：在 oo_.mom 显示并存储中间估计结果。默认值：不设置。

SMM 特定选项

burnin=INTEGER：模拟开始时舍弃的期数。默认值：500。

bounded_shock_support：将模拟的冲击微调至±2 个标准偏差。默认值：不设置。

seed=INTEGER：模拟中常用的种子。默认值：24051986。

simulation_multiple=INTEGER：模拟的数据长度的倍数。默认值：7。

GMM 特定选项

analytic_standard_errors：使用与估计参数有关的矩的解析导数计算标准误差。默认值：不设置，即标准误差是使用双边有限差分法计算的，参见 *se_tolx*。

一般选项

dirname=FILENAME：存储 estimation 输出的路径。更多信息参见 *dirname*。默认值：<mod_file>。

graph_format=FORMAT：指定保存到磁盘的图形文件格式。更多信息参见 *graph_format*。默认值：eps。

nodisplay：参见 *nodisplay*。默认值：不设置。

nograph：参见 *nograph*。默认值：不设置。

noprint：参见 *noprint*。默认值：不设置。

plot_priors=INTEGER：控制先验图的绘制。更多信息参见 *plot_priors*。默认值：1，即绘制先验图。

prior_trunc=DOUBLE：更多信息参见 *prior_trunc*。默认值：1e-10.

tex：参见 *tex*。默认值：不设置。

数据选项

first_obs=INTEGER：参见 *first_obs*。默认值：1。

nobs=INTEGER：参见 *nobs*。默认值：考虑所有观测值。

prefilter=INTEGER：取值为 1 意味着估计程序将根据经验平均值对每个数据序列去均值，并根据理论平均值对每个模型矩进行去均值。更多信息参见 *prefilter*。默认值：0，即没有提前滤波。

logdata：参见 *logdata*。默认值：不设置。

xls_sheet=QUOTED_STRING：参见 *xls_sheet*。

xls_range=RANGE：参见 *xls_range*。

最优选项

huge_number=DOUBLE：参见 *huge_number*。默认值：1e7。

mode_compute=INTEGER|FUNCTION_NAME：参见 *mode_compute*。默认值：13，如果存在 Matlab 最优化工具箱或者 Octave 最优化包，即 lsqnonlin，否则取值为 4，即 csminwel。

additional_optimizer_steps=[INTEGER|FUNCTION_NAME,INTEGER|FUNC

TION_NAME,...]：mode_compute 之后运行的附加最小化算法向量。如果设置了 *verbose* 选项，附加的估计结果将以前缀为 *verbose*_的形式保存到 oo_.mom。默认值：没有附加的最优化迭代。

optim=(NAME,VALUE,...)：参见 *optim*。

silent_optimizer：参见 *silent_optimizer*。默认值：不设置

数值算法选项

aim_solver：参见 *aim_solver*。默认值：不设置。

k_order_solver：参见 *k_order_solver*。默认值：禁用一阶和二阶，适用于三阶及以上。

dr=OPTION：参见 *dr*。默认值：default，即广义 Schur 分解。

dr_cycle_reduction_tol=DOUBLE：参见 *dr_cycle_reduction_tol*。默认值：1e-7.

dr_logarithmic_reduction_tol=DOUBLE：参见 *dr_logarithmic_reduction_tol*。默认值：1e-12。

dr_logarithmic_reduction_maxiter=INTEGER：参见 *dr_logarithmic_reduction_maxiter*。默认值：100。

lyapunov=OPTION：参见 *lyapunov*。默认值：default，即基于 Bartlets-Stewart 算法。

lyapunov_complex_threshold=DOUBLE：参见 *lyapunov_complex_threshold*。默认值：1e-15。

lyapunov_fixed_point_tol=DOUBLE：参见 *lyapunov_fixed_point_tol*。默认值：1e-10。

lyapunov_doubling_tol=DOUBLE：参见 *lyapunov_doubling_tol*。默认值：1e-16。

sylvester=OPTION：参见 *sylvester*。默认值：default，即使用 gensylv。

sylvester_fixed_point_tol=DOUBLE：参见 *sylvester_fixed_point_tol*。默认值：1e-12。

qz_criterium=DOUBLE：参见 *qz_criterium*。默认值：0.999999，因为假定观测变量是弱稳态的。

qz_zero_threshold=DOUBLE：参见 *qz_zero_threshold*。默认值：1e-6。

schur_vec_tol=DOUBLE：转移矩阵的 Schur 分解中查找非平稳变量的可容忍度。默认值：1e-11。

mode_check：围绕每个待估参数计算的最小值，通过矩距目标函数绘制单变量切片，这有助于诊断优化问题。默认值：不设置。

mode_check_neighbourhood_size=DOUBLE：参见 *mode_check_neighbourhood_size*。默认值：0.5。

mode_check_symmetric_plots=INTEGER：参见 *mode_check_symmetric_plots*。默认值：1。

mode_check_number_of_points=INTEGER：参见 *mode_check_number_of_points*。默认值：20。

输出

method_of_moments 将用户选项存储在全局工作空间中名为 options_mom_的结构中。运行估计之后，参数 M_.params 与冲击 M_.Sigma_e 和测量误差 M_.H 的协方差矩阵被设置为最小化二次矩距离目标函数的参数。估计结果存储在 oo_.mom 结构中，包含以下字段：

MATLAB/Octave variable:oo_.mom.data_moments

变量由 method_of_moments 命令设置。存储所选数据经验矩的平均值。计算平均值时，由于超前/滞后或缺失数据而导致的 NaN 值被忽略。向量维度等于正交条件的数量。

MATLAB/Octave variable:oo_.mom.m_data

变量由 method_of_moments 命令设置。存储每个时间点的选定经验矩。由于超前/滞后或缺失数据而导致的 NaN 值被相应的时刻平均值替换。矩阵的维度时间周期乘以正交条件的数量。

MATLAB/Octave variable:oo_.mom.Sw

变量由 method_of_moments 命令设置。存储当前使用的加权矩阵 Cholesky 分解。方阵维度等于正交条件的数量。

MATLAB/Octave variable:oo_.mom.model_moments

变量由 method_of_moments 命令设置。在给定当前参数猜测的情况下存储隐含的选定模型矩。模型矩是从 GMM 的剪枝状态空间系统中以封闭形式计算的，而对于 SMM，这些是基于模拟数据的平均值。向量维度等于正交条件的数量。

MATLAB/Octave variable:oo_.mom.Q

变量由 method_of_moments 命令设置。存储二阶矩距离目标函数的标量值。

MATLAB/Octave variable:oo_.mom.model_moments_params_derivs

变量由 method_of_moments 命令设置。存储经分析计算的模型矩相对于估计参数的导数的雅可比矩阵。仅用于具有 *analytic_standard_errors* 的 GMM。矩阵的维度等于正交条件的数量乘以待估参数的数量。

MATLAB/Octave variable:oo_.mom.gmm_stage_*_mode

MATLAB/Octave variable:oo_.mom.smm_stage_*_mode

MATLAB/Octave variable:oo_.mom.verbose_gmm_stage_*_mode

MATLAB/Octave variable:oo_.mom.verbose_smm_stage_*_mode

使用 GMM 或 SMM 估计时变量由 method_of_moments 命令设置。存储阶段 1、2、……的估计值。这些结构包含以下字段：

- measurement_errors_corr：估计两个测量误差之间的相关性；
- measurement_errors_std：测量误差的估计标准偏差；
- parameters：估计的模型参数；
- shocks_corr：估计两个结构性冲击之间的相关性；
- shocks_std：结构性冲击的估计标准差。

如果设置了 *verbose* 选项，以 verbose_为前缀的附加字段将被保存于所有 *additional_optimizer_steps*。

MATLAB/Octave variable:oo_.mom.gmm_stage_*_std_at_mode

MATLAB/Octave variable:oo_.mom.smm_stage_*_std_at_mode

MATLAB/Octave variable:oo_.mom.verbose_gmm_stage_*_std_at_mode

MATLAB/Octave variable:oo_.mom.verbose_smm_stage_*_std_at_mode

使用 GMM 或 SMM 估计时由 method_of_moments 命令设置的变量。存储阶段 1、2、……的估计标准误差。这些结构包含以下字段：

- measurement_errors_corr：两个测量误差之间估计相关性的标准误差；
- measurement_errors_std：测量误差的估计标准偏差的标准误差；
- parameters：估计模型参数的标准误差；
- shocks_corr：估计两个结构性冲击之间相关性的标准误差；
- shocks_std：结构性冲击的估计标准差的标准误差。

如果设置了 *verbose* 选项，以 verbose_为前缀的附加字段将被保存用于所有 *additional_optimizer_steps*。

MATLAB/Octave variable:oo_.mom.J_test

变量由 method_of_moments 命令设置。检验统计量的值保存到名为 j_stat 的字段中，自由度保存到名为 degree_freedom 的字段中，检验统计量的 p 值保存到名为 p_val 的字段中的结构。

4.17 模型比较

Command:model_comparison FILENAME[(DOUBLE)]...;

Command:model_comparison(marginal_density=ESTIMATOR)FILENAME[(DOUBLE)]...;

计算优势比并估计模型集合上的后验密度（参见 Koop，2003，第 1 章）。模型的先验可以指定为 *DOUBLE*，否则假设所有模型的先验是一致的。与常用计量经济学相反，要比较的模型不需要嵌套。然而，由于后验优势比的计算是一种贝叶斯方法，因此不支持用最

大似然估计模型的比较。

重要的是要记住，此类模型比较只在适当的先验下有效。如果先验没有集成所有的比较模型，则比较无效。如果不满足 Blanchard 和 Kahn 条件（模型的不稳定性或不确定性），或者未定义某些参数空间区域的确定性稳态（或 Dynare 无法找到），先验质量的一部分被暗中截断，则可能出现这种情况。比较的边缘密度应该通过有效先验质量重新归一化，但这不是由 Dynare 完成的：用户有责任确保模型比较基于适当的先验。注意，出于显而易见的原因，如果比较的边际密度基于拉普拉斯近似值，则不是问题。

选项

marginal_density=ESTIMATOR：指定用于计算边际数据密度的估计程序。*ESTIMATOR* 可以取以下两个值中的一个：`laplace` 拉普拉斯估计量或 Geweke（1999）提出的 `modifiedharmonicmean` 修正谐波平均估计量。默认值：`laplace`。

输出

结果存储在 `oo_.Model_Comparison`，如下所述。

示例

```
model_comparison my_model(0.7) alt_model(0.3);
```

此示例将 70%的先验归因于 `my_model` 和 30%的先验归因于 `alt_model`。

MATLAB/Octave variable:oo_.Model_Comparison

`model_compare` 命令设置的变量。字段形式：

```
oo_.Model_Comparison.FILENAME.VARIABLE_NAME
```

其中 FILENAME 是模型的文件名，VARIABLE_NAME 是以下之一：

- `Prior`：模型上的（规范化）先验密度；
- `Log_Marginal_Density`：边缘数据密度的对数；
- `Bayes_Ratio`：模型的边缘数据密度与第一个声明模型之比；
- `Posterior_Model_Probability`：每个模型的后验概率。

4.18 冲击分解

Command:shock_decomposition[VARIABLE_NAME]...;

Command:shock_decomposition(OPTIONS...)[VARIABLE_NAME]...;

基于卡尔曼平滑器计算给定样本的历史冲击分解，即将内生变量的历史偏差从各自的稳态值分解为来自各种冲击的贡献。提供的 `variable-names` 控制已绘制分解的变量。注意，这个命令必须在 `estimate`（待估模型）或随机 `simul`（校准模型）之后。

选项

parameter_set=OPTION：指定运行平滑器的参数集。*OPTION* 的可能值是：

- `Calibration`；
- `prior_mode`；

- prior_mean;
- posterior_mode;
- posterior_mean;
- posterior_median;
- mle_mode。

注意，后续命令（如 stoch_simul）使用的参数集将设置为指定的 parameter_set。默认值：如果 Metropolis 采样算法已运行，则选择 posterior_mean，如果 MLE 已运行，则选择 mle_mode。

datafile=FILENAM：参见 *datafile*。在校准模型计算冲击分解时非常有用。

first_obs=INTEGER：参见 *first_obs*。

nobs=INTEGER：参见 *nobs*。

prefilter=INTEGER：参见 *prefilter*。

loglinear：参见 *loglinear*。

diffuse_kalman_tol=DOUBLE：参见 *diffuse_kalman_tol*。

diffuse_filter：参见 *diffuse_filter*。

xls_sheet=NAME：参见 *xls_sheet*。

xls_range=RANGE：参见 *xls_range*。

use_shock_groups[=NAME]：使用由字符串定义的冲击分组而不是分解中的单个冲击。冲击组在 *shock_groups* 块中定义。如果没有给出组名，则假定为 default。

colormap=VARIABLE_NAME：控制冲击分解图的 colormap。VARIABLE_NAME 必须是事先声明的 MATLAB/Octave 变量的名称，其值将传递给 MATLAB/Octave 的 colormap 函数（可接受值的列表参见 MATLAB/Octave 手册）。

nograph：参见 *nograph*。仅在 *shock_decomposition* 命令内禁止显示和创建，但不影响其他命令。绘制图形参见 *plot_shock_decomposition*。

init_state=BOOLEAN：如果等于 0，则冲击分解的计算条件是周期 0 中的平滑状态变量，即使用从周期 1 开始的平滑冲击。如果等于 1，则冲击分解的计算条件是周期 1 中的平滑状态变量。默认值：0。

with_epilogue：如果设置，还要计算在 epilogue 块中声明变量的分解（参见 4.22 尾声变量）。

输出

MATLAB/Octave variable:oo_.realtime_shock_decomposition

结果存储在字段 oo_.shock_decomposition 中，这是一个三维数组。第一个维度包含 M_.endo_nbr 内生变量；第二个维度将各个冲击的贡献存储在 M_.exo_nbr 第一列。M_.exo_nbr+1 列存储初始条件的贡献，而 M_.exo_nbr+2 列存储各自内生变量与

其稳态偏差的平滑值，即减去平均值和趋势；第三个维度存储时间段。变量和冲击都按照声明顺序存储，即分别是 M_.endo_names 和 M_.exo_names。

Block:shock_groups;

Block:shock_groups(OPTIONS...);

为了分解冲击，可以重新进行分组冲击。冲击组的组成写在一个由 shock_groups 和 end 分隔的块中。每行将一组冲击定义为一组外生变量：

```
SHOCK_GROUP_NAME   = VARIABLE_1[[,]VARIABLE_2[,]...];
'SHOCK GROUP NAME' = VARIABLE_1[[,]VARIABLE_2[,]...];
```

选项

name=NAME：为以下冲击组定义指定名称。可以在一个模型文件中使用多个 shock_groups 模块，每个分组由不同的名称标识。该名称必须依次用于 shock_decomposition 命令。如果没有给出名称，则使用 default。

示例

```
varexo e_a, e_b, e_c, e_d;
...

shock_groups(name=group1);
supply=e_a,e_b;
'aggregate demand' = e_c, e_d;
end;

shock_decomposition(use_shock_groups=group1);
```

此例定义一个包含一组供求冲击，名为 group1 的冲击组，并对两个组实施冲击分解。

Command:realtime_shock_decomposition[VARIABLE_NAME]...;

Command:realtime_shock_decomposition(OPTIONS...)[VARIABLE_NAME]...;

基于卡尔曼平滑器计算给定样本的实时历史冲击分解。每个周期T=[presample,...,nobs]递归计算三个对象：

- 对$t = [1, \ldots, T]$的实时历史冲击分解$Y(t|T)$，即没有观察$[T + 1, \ldots,$ nobs]的数据。这导致为每个附加数据点计算标准冲击分解，在 presample 后可用；
- 对于$k = [1, \ldots, forecast]$预测冲击分解$Y(T + k|T)$，即对每个$T$做出的$k$步超前预测在其冲击贡献中分解；
- 在实时历史冲击分解与预测冲击分解之间的实时条件冲击分解区别。如果 Vintage 等于 0，则计算在周期T内实现的冲击的影响，即分解$Y(T|T) - Y(T|T - 1)$。

换句话说，它通过分解卡尔曼滤波的更新步骤，进行从$T-1$到T的 1 周期超前冲击分解。如果 vintage>0 且小于 nobs，则分解预测修正$Y(T+k|T+k)-Y(T+k|T)$。

与 *shock_decomposition* 一样，它将内生变量的历史偏差从它们各自的稳态值分解为来自各种冲击的贡献。提供的 variable_names 决定了对哪些变量绘制分解。

注意，此命令必须在 estimate（估计模型）或 stoch_simul（校准模型）后出现。

选项

parameter_set=OPTION：可能的值参见 *parameter_set*。

datafile=FILENAME：参见 *datafile*。

first_obs=INTEGER：参见 *first_obs*。

nobs=INTEGER：参见 *nobs*。

use_shock_groups[=NAME]：参见 *use_shock_groups*。

colormap=VARIABLE_NAME：参见 *colormap*。

nograph：参见 *nograph*。只计算冲击分解并将其存储在 oo_.realtime_shock_decomposition、oo_.conditional_shock_decomposition 和 oo_.realtime_forecast_shock_decomposition，但没有绘制图（参见 *plot_shock_decomposition*）。

presample=INTEGER：计算递归实时冲击分解的数据点，即对于T=[presample+1...nobs]。

forecast=INTEGER：计算最多$T+k$个周期的冲击分解，即获得对k步超前预测的冲击贡献。

save_realtime=INTEGER_VECTOR：选择保存完整实时冲击分解年份。默认值：0。

fast_realtime=INTEGER

fast_realtime=[INTEGER1:INTEGER2]

fast_realtime=[INTEGER1 INTEGER2...]：只对指定整数（向量）提供的数据年份运行平滑器

with_epilogue：参见 *with_epilogue*。

输出

MATLAB/Octave variable:oo_.realtime_shock_decomposition

存储实时历史分解结果的结构。字段是三维数组，前两个维度等于 *oo_.shock_decomposition*。第三个维度存储时间段，因此大小为 T+forecast。字段形式：

```
oo_.realtime_shock_decomposition.OBJECT
```

其中 OBJECT 是以下之一：

- pool：存储合并分解，即对于每个实时冲击分解终止周期T=[presamp

le，...，nobs]，它收集最后一个周期的分解$Y(T|T)$（另见 *plot_shock_decomposition*）。第三维度大小是 nobs+forecast；

- time_*：如果使用 save_realtime，则存储实时历史冲击分解的年份。例如，如果 save_realtime=[5]并且 forecast=8，第三个维度的大小为 13。

MATLAB/Octave variable:oo_.realtime_conditional_shock_decomposition

存储实时条件分解结果的结构。字段形式：

```
oo_.realtime_conditional_shock_decomposition.OBJECT
```

其中 OBJECT 是以下之一：

- pool：存储合并的实时条件冲击分解，即对于终止时段T=[presample，...，nobs]收集$Y(T|T) - Y(T|T-1)$的分解。第三个维度大小是 nobs；
- time_*：对于$t = [T \ldots T+k]$时存储k步条件预测冲击分解$Y(t|T+k)$的年份。参见 *vintage*。第三个维度大小是 1+forecast。

MATLAB/Octave variable:oo_.realtime_forecast_shock_decomposition

存储实时预测分解结果的结构。字段形式：

```
oo_.realtime_forecast_shock_decomposition.OBJECT
```

其中 OBJECT 是以下之一：

- pool：存储冲击对 1 步超前预测的影响的合并实时预测分解结果，即$Y(T|T-1)$；
- time_*：对于$t = \cdots + k]$，存储k步样本外预测冲击分解的年份，即$Y(t|T)$。参见 *vintage*。

Command:plot_shock_decomposition[VARIABLE_NAME]...;

Command:plot_shock_decomposition(OPTIONS...)[VARIABLE_NAME]...;

绘制由 shock_decomposition 或 realtime_shock_decomposition 计算的历史冲击分解。因此，它必须在这些命令之后。提供的 variable_names 决定了绘制分解的变量。进一步注意，与大多数 Dynare 命令不同，在每次调用 plot_shock_decomposition 之前，下面指定的选项都会被默认值覆盖。因此，如果要在后续调用 plot_shock_decomposition 时重新使用选项，则必须再次传递给命令。

选项

use_shock_groups[=NAME]：参见 *use_shock_groups*。

colormap=VARIABLE_NAME：参见 *colormap*。

nodisplay：参见 *nodisplay*。

nograph：参见 *nograph*。

graph_format=FORMAT graph_format=(FORMAT,FORMAT...)：参见 *graph_format*。

detail_plot：使用子图绘制冲击贡献，每次冲击（或一组冲击）。默认值：未激活。

interactive：MATLAB 环境添加 uimenus 以获取详细的组图。默认值：未激活。

screen_shocks：对于大型模型（即对于具有超过 16 次冲击的模型），仅绘制对所选择的 variable_name 具有最大历史贡献的冲击。历史贡献按所有历史贡献的平均绝对值排序。

steadystate：如果通过，则冲击分解图中零线的y轴值转换为稳态水平。默认值：未激活。

type=qoq|yoy|aoa：对于季度数据，有效的参数是：季度环比图 qoq，同比增长率图 yoy，年化变量 aoa，即绘制每年最后一个季度的值。默认值：空，即标准周期图（季度数据的 qoq）。

fig_name=STRING：指定附加到 plot_shock_decomposition 设置的默认图形名称的用户定义关键字。这可以避免在连续调用 plot_shock_decomposition 的情况下覆盖绘图。

write_xls：冲击分解保存到主目录的 Excel 文件，命名为 FILENAME_shock_decomposition_TYPE_FIG_NAME.xls，此选项要求将系统配置为能够编写 Excel 文件。[⑧]

realtime=INTEGER：以哪种冲击分解来绘制。INTEGER 可以采用以下值：

0：标准历史冲击分解。参见 *shock_decomposition*；

1：实时历史冲击分解。参见 *realtime_shock_decomposition*；

2：条件实时冲击分解。参见 *realtime_shock_decomposition*；

3：实时预测冲击分解。参见 *realtime_shock_decomposition*。

如果没有年份，即 vintage=0，则绘制来自 *realtime_shock_decomposition* 的合并对象，否则将绘制相应的年份。默认值：0

vintage=INTEGER：在$[presam \ldots, nobs]$一个特定的数据年份，通过 *realtime* 选项选择用于绘制 *realtime_shock_decomposition* 的结果。如果选择标准历史冲击分解（realtime=0），则 vintage 将不起作用。如果 vintage=0，将绘制来自 *realtime_shock_decomposition* 的合并对象，如果 vintage>0，则在下列情况下绘制历史年份T=vintage 的冲击分解：

- realtime=1：对于$t \ldots\ldots, T]$的全历史年份冲击分解$Y(t|T)$；
- realtime=2：来自T的条件预测冲击分解，即绘图$Y(T+j|T+j)$和以T=vintage 为条件得到数据$Y(T+j)$所需的冲击贡献，其中j=[0,...,forecast]；
- realtime=3：绘制来自T的无条件预测冲击分解，即$Y(T+j|T)$，其中T=vinta

⑧ 为了防止没有安装 Excel，以下链接可能有帮助 https://mathworks.com/matlabcentral/fileexchange/38591-xlwrite–generate-xls-x–files-without-excel-on-mac-linux-win。

ge 且*j*=[0,...,forecast]。

默认值：0

plot_init_date=DATE：如果通过，则使用 plot_init_date 作为初始时间段绘制分解图。默认值：估计中的首个观察值。

plot_end_date=DATE：如果通过，则使用 plot_end_date 作为最后一个时期绘制分解图。默认值：估计中的最后观察值。

diff：如果通过，则绘制变量列表的首个差异的分解。如果与 *flip* 结合使用，则首先应用 diff 运算符。默认值：未激活。

flip：如果通过，则绘制与变量列表相反的分解图。如果与 *diff* 结合使用，则首先应用 diff 运算符。默认值：未激活。

max_nrows：详细冲击分解图的子图布局中的最大行数。列始终为 3。默认值：6。

with_epilogue：参见 *with_epilogue*。

init2shocks

init2shocks=NAME：使用 *init2shocks* 模块中包含的信息，将初始条件归因于冲击。可以明确给出模块的名称，否则默认为 default 模块。

Block:init2shocks;

Block:init2shocks(OPTIONS...);

该模块在冲击分解中提供了将内生变量的初始条件归因于外生变量贡献的可能性。例如，在 AR(1)过程，初始条件对过程变量的贡献可以自然地分配给过程的创新。

模块的每一行都应具有以下语法：

```
VARIABLE_1[,]VARIABLE_2;
```

其中 VARIABLE_1 是一个内生变量，其初始条件将归因于外生 VARIABLE_2。

给定 init2shocks 选项时，模块包含的信息由 *plot_shock_decomposition* 命令使用。

选项

name=NAME：指定模块的名称，可从 plot_shock_decomposition 引用该名称，以便多个此类模块共存于单个模型文件中。如果未指定名称，则默认为 default。

示例

```
var y y_s R pie dq pie_s de A y_obs pie_obs R_obs;
varexo e_R e_q e_ys e_pies e_A;
...


model;
  dq=rho_q*dq(-1)+e_q;
```

```
  A=rho_A*A(-1)+e_A;
  ...
end;


...


init2shocks;
  dq e_q;
  A e_A;
end;


shock_decomposition(nograph);


plot_shock_decomposition(init2shocks) y_obs R_obs pie_obs dq d
e;
```

示例 dq 和 A 的初始条件分别归属于 e_q 和 e_A。

Command:initial_condition_decomposition[VARIABLE_NAME]...;

Command:initial_condition_decomposition(OPTIONS...)[VARIABLE_NAM E]...;

计算并绘制状态变量平滑初始条件的效果分解图。提供的 variable_names 控制为哪些变量绘制分解图。

进一步注意，与大多数 Dynare 命令不同，在每次调用 initial_condition_decom position 之前，下面指定的选项都会被它们的默认值覆盖。因此，如果您想在对 initi al_condition_decomposition 的后续调用中重新使用某个选项，则必须再次将其传递给命令。

选项

colormap=VARIABLE_NAME：参见 *colormap*.

nodisplay：参见 *nodisplay*。

graph_format=FORMAT graph_format=(FORMAT,FORMAT...)：参见 *graph_ format*。

detail_plot：使用子图绘制冲击贡献，每个冲击（或冲击组）对应一个。默认值：未激活。

steadystate：如果通过，则冲击分解图中零线的y轴值将转换为稳态水平。默认值：未激活。

type=qoq|yoy|aoa：对于季度数据，有效参数是：季度环比图的 qoq，同比增长率图的 yoy，年化变量的 aoa，即每年最后一个季度的值。默认值：空，即标准周期图（季度数据的 qoq）。

fig_name=STRING：指定要附加到 plot_shock_decomposition 设置的默认图形名称的用户定义关键字。这可以避免在顺序调用 plot_shock_decomposition 的情况下覆盖绘图。

write_xls：冲击分解保存到主目录中的 Excel 文件，命名为 FILENAME_shock_decomposition_TYPE_FIG_NAME.xls，此项要求将系统配置为能够编写 Excel 文件。⑨

plot_init_date=DATE：如果通过，则使用 plot_init_date 作为初始时间段绘制分解图。默认值：估计中的首个观察值。

plot_end_date=DATE：如果通过，则使用 plot_end_date 作为最后一个时期绘制分解图。默认值：估计中的最后一次观察值。

diff：如果通过，则绘制变量列表的首个差异的分解。如果与 *flip* 结合使用，则首先应用 diff 运算符。默认值：未激活。

flip：如果通过，则绘制与变量列表相反的分解图。如果与 *diff* 结合使用，则首先应用 diff 运算符。默认值：未激活。

Command:squeeze_shock_decomposition[VARIABLE_NAME]...;

对于大型模型，冲击分解（尤其是实时分解的各种设置）存储的信息量可能会变得巨大。此命令允许以两种可能的方式压缩此信息：

- 自动（默认）：在运行此命令后，只有使用 plot_shock_decomposition 明确要求绘图的变量才会将分解结果留在 oo_ 中；
- 如果变量列表传递给命令，则在运行后，只有这些变量的分解会留在 oo_。

4.19 校准平滑

Dynare 还可以在校准模型上运行平滑器：

Command:calib_smoother[VARIABLE_NAME]...;

Command:calib_smoother(OPTIONS...)[VARIABLE_NAME]...;

计算校准模型上的平滑变量（以及可能的滤波变量）。必须提供数据文件，使用 varobs 声明可观察变量。平滑器基于模型的一阶近似。默认情况下计算平滑变量和冲击，结果存储在 oo_.SmoothedVariables 和 oo_.SmoothedShocks 中。它还填充 oo_.UpdatedVariables。

选项

datafile=FILENAME：参见 *datafile*。

⑨ 为了防止没有安装 Excel，以下链接可能有帮助 https://mathworks.com/matlabcentral/fileexchange/38591-xlwrite–generate-xls-x–files-without-excel-on-mac-linux-win。

filtered_vars：触发滤波变量的计算，更多细节参见 *filtered_vars*。

filter_step_ahead=[INTEGER1:INTEGER2]：参见 *filter_step_ahead*。

prefilter=INTEGER：参见 *prefilter*。

parameter_set=OPTION：可能的值参见 *parameter_set*。默认值：calibration。

loglinear：参见 *loglinear*。

first_obs=INTEGER：参见 *first_obs*。

filter_decomposition：参见 *filter_decomposition*。

filter_covariance：参见 *filter_covariance*。

smoother_redux：参见 *smoother_redux*。

kalman_algo=INTEGER：参见 *kalman_algo*。

diffuse_filter=INTEGER：参见 *diffuse_filter*。

diffuse_kalman_tol=DOUBLE：参见 *diffuse_kalman_tol*。

xls_sheet=NAME：参见 *xls_sheet*。

xls_range=RANGE：参见 *xls_range*。

4.20 预测

校准模型使用 forecast 命令预测，估计模型使用 estimate 命令的 forecast 选项。对未来内生变量的给定约束路径的校准或估计模型的预测也可计算。这是通过 DSGE 模型的简化形式表示，找到匹配受限路径的结构冲击完成的，可用 conditional_forecast、conditional_forecast_paths 和 plot_conditional_forecast 实现。

最后，使用 bvar_forecast 命令预测贝叶斯 VAR。

Command:forecast[VARIABLE_NAME...];

Command:forecast(OPTIONS...)[VARIABLE_NAME...];

计算从任意初始点开始的随机模型的模拟。当模型还包含确定性外生冲击时，模拟是有条件地计算知道确定性外生变量的未来值的代理。必须在 stoch_simul 之后调用 forecast。

forecast 绘制了内生变量的轨迹。当命令后面跟着变量名列表时，只绘制那些变量。围绕平均轨迹绘制 90%置信区间。使用选项 conf_sig 更改置信区间的水平。

选项

periods=INTEGER：预测期数。默认值：5。

conf_sig=DOUBLE：置信区间的显著性水平。默认值：0.90

nograph：参见 *nograph*。

nodisplay：参见 *nodisplay*。

graph_format=FORMAT graph_format=(FORMAT,FORMAT...)：参见 *graph_*

format=FORMAT。

初始值

forecast 计算将 histval 指定的值作为初始值（参见 *histval*）。如果不存在 histval 模块，则初始值是 initval 指定的值。当 initval 后跟命令 steady 时，初始值为稳态（参见 *steady*）。

输出

结果存储在 oo_.forecast 中，如下所述。

示例

```
varexo_det tau;

varexo e;
...
shocks;
var e; stderr 0.01;
var tau;
periods 1:9;
values -0.15;
end;

stoch_simul(irf=0);

forecast;
```

MATLAB/Octave variable:oo_.forecast

变量由 forecast 命令设置，如果与 forecast 选项和 ML 一起使用，或没有计算 Metropolis-Hastings（此时预测是针对后验众数计算的），则由 estimate 命令设置。字段形式：

```
oo_.forecast.FORECAST_MOMENT.VARIABLE_NAME
```

其中 FORECAST_MOMENT 是以下之一：

- HPDinf：由于参数不确定性产生了 90%HPD 预测区间[10]的下限，但忽略了测量误差对观测变量的影响。在 ML 的情况下，它保留了置信区间的下界；
- HPDsup：由于参数不确定性产生了 90%HPD 预测区间的上限，但忽略了测量误差对观测变量的影响。在 ML 的情况下，它保留了置信区间的上界；
- HPDinf_ME：由于参数不确定性和测量误差导致观测变量预测的 90%H

⑩ 参见选项 *conf_sig* 来改变 HPD 区间的大小。

PD 区间的[11]下限；

- `HPDsup_ME`：由于参数不确定性和测量误差导致观测变量预测的 90%HPD 区间的上限；
- `Mean`：预测后验分布的均值。

MATLAB/Octave variable:oo_.PointForecast

如果与 `forecast` 选项一起使用，并且使用了 `mh_replic>0` 或 `load_mh_file` 选项，变量由 `estimation` 命令设置。

包含考虑到参数和冲击不确定性的预测分布。字段形式：

```
oo_.PointForecast.MOMENT_NAME.VARIABLE_NAME
```

MATLAB/Octave variable:oo_.MeanForecast

如果与 `forecast` 选项一起使用，并且使用了 `mh_replic>0` 或 `load_mh_file` 选项，变量由 `estimation` 命令设置。

包含平均化冲击不确定性的预测分布。故预测的分布仅代表参数不确定性。字段形式：

```
oo_.MeanForecast.MOMENT_NAME.VARIABLE_NAME
```

Command:conditional_forecast`(OPTIONS...);`

用于计算对于给定了某些未来内生变量约束路径的估计或校准模型的预测。这是用 DSGE 模型的简化式一阶状态空间表示法完成的，即找到与受限路径相匹配的结构冲击。考虑增广状态空间表示，将预定和非预定的变量都堆积成一个向量y_t：

$$y_t = Ty_{t-1} + R\varepsilon_t$$

y_t和ε_t分别表示控制变量和非控制变量组成的向量，不失一般性，假设受约束的内生变量和受控冲击首先出现：

$$\begin{pmatrix} y_{c,t} \\ y_{u,t} \end{pmatrix} = \begin{pmatrix} T_{c,c} & T_{c,u} \\ T_{u,c} & T_{u,u} \end{pmatrix} \begin{pmatrix} y_{c,t-1} \\ y_{u,t-1} \end{pmatrix} + \begin{pmatrix} R_{c,c} & R_{c,u} \\ R_{u,c} & R_{u,u} \end{pmatrix} \begin{pmatrix} \varepsilon_{c,t} \\ \varepsilon_{u,t} \end{pmatrix}$$

其中矩阵T和R与内生变量和创新变量组成的向量划分一致。只要矩阵$R_{c,c}$是满秩的方阵（必要条件是自由变化的内生变量数目匹配自由变化的更新变量数目），给定$y_{c,t}$、$\varepsilon_{u,t}$和y_{t-1}就可以根据第一组方程求得$\varepsilon_{c,t}$：

$$\varepsilon_{c,t} = R_{c,c}^{-1}\left(y_{c,t} - T_{c,c}y_{c,t} - T_{c,u}y_{u,t} - R_{c,u}\varepsilon_{u,t}\right)$$

并且$y_{u,t}$可通过计算第二组方程替换：

$$y_{u,t} = T_{u,c}y_{c,t-1} + T_{u,u}y_{u,t-1} + R_{u,c}\varepsilon_{c,t} + R_{u,u}\varepsilon_{u,t}$$

通过迭代两组方程的，可以根据内生变量子集的路径建立一个系统中所有内生变量的预测。如果给定了自由创新变量$\varepsilon_{u,t}$的分布（即其中一些具有正方差），则通过绘制不同的自由创新序列重复这一操作（重复次数由下文描述的 `replic` 选项控制）。结果是非控制内生变量$y_{u,t}$的预测分布，Dynare 将用它来报告点条件预测周围的置信区间。

11 参见选项 *conf_sig* 来改变 HPD 区间的大小。

有一些事情需要注意：首先，对于所有不受控制的时期，控制外生变量被设定为 0。这意味着在不受控制的时期内，这些外生变量不会产生预测不确定性。其次，利用一阶状态空间解，即使执行高阶近似，条件预测也将基于一阶近似。由于受控的外生变量是基于简化式模型确定的（即求解期望之后），从模型代理人的角度来看，它们是不可预见的冲击。也就是说，代理人预期内生变量会回到各自的稳态水平，但在每个时期都会因内生变量根据预先确定的（不可预期的）路径形成的冲击感到意外；最后，如果结构创新之间是彼此相关的，那么由于校准或估计的协方差矩阵具有非零对角线元素，条件预测的结果将取决于创新的排序（如 varexo 之后声明）。与 VAR 模型一样，Cholesky 分解用于分解协方差矩阵并识别正交脉冲。除非对 Cholesky 分解隐含的识别约束感到满意，否则最好在模型模块中声明相关性（明确施加识别约束）。

此命令必须在 estimation 或 stoch_simul 之后调用。

使用 *conditional_forecast_paths* 模块提供受约束的内生列表以及列表中受约束的未来路径。选项 controlled_varexo 用于指定匹配以生成约束路径的结构冲击。

使用 *plot_conditional_forecast* 来绘制结果图。

选项

parameter_set=OPTION：可能值参见 *parameter_set*。无默认值，必选项。

controlled_varexo=(VARIABLE_NAME...)：指定用作控制变量的外生变量。无默认值，必选项。

periods=INTEGER：预测期数。默认值：40。periods 不可能比受限的期数短。

replic=INTEGER：模拟次数。默认值：5000。

conf_sig=DOUBLE：置信区间的显著性水平。默认值：0.80

输出

结果存储在 oo_.conditional_forecast 中，如下所述。

示例

```
var y a;
varexo e u;
...
estimation(...);

conditional_forecast_paths;
var y;
periods 1:3,4:5;
values 2,5;
var a;
```

```
periods 1:5;
values 3;
end;

conditional_forecast(parameter_set=calibration,controlled_var
exo=(e,u),replic=3000);

plot_conditional_forecast(periods=10) a y;
```

MATLAB/Octave variable:oo_.conditional_forecast.cond

变量由 conditional_forecast 命令设置，存储条件预测。字段是存储稳态（时间0）和后续 periods 预测时段（period+1）×1 的向量。字段形式：

```
oo_.conditional_forecast.cond.FORECAST_MOMENT.VARIABLE_NAME
```

其中 FORECAST_MOMENT 是以下之一：

- Mean：条件预测分布的均值；
- ci：条件预测分布的置信区间。大小对应 conf_sig。

MATLAB/Octave variable:oo_.conditional_forecast.uncond

变量由 conditional_forecast 命令设置，存储无条件预测。字段形式：

```
oo_.conditional_forecast.uncond.FORECAST_MOMENT.VARIABLE_NAME
```

MATLAB/Octave variable:forecasts.instruments

变量由 conditional_forecast 命令设置，存储外生工具的名称。

MATLAB/Octave variable:oo_.conditional_forecast.controlled_variables

变量由 conditional_forecast 命令设置，按声明顺序存储受约束的内生变量位置。

MATLAB/Octave variable:oo_.conditional_forecast.controlled_exo_variables

变量由 conditional_forecast 命令设置，存储作为基于条件预测的受控外生变量的值，以实现受约束的内生变量。字段是[number of constrained periods]×1 的向量，形式为：

```
oo_.conditional_forecast.controlled_exo_variables.FORECAST_MO
MENT.SHOCK_NAME
```

MATLAB/Octave variable:oo_.conditional_forecast.graphs

变量由 conditional_forecast 命令设置，存储用于生成条件预测图的信息。

Block:conditional_forecast_paths;

在调用 conditional_forecast 之前描述内生约束的路径。语法类似于冲击中的确

定性冲击，参见 conditional_forecast 示例。

模块语法与 shocks 模块中的确定性冲击相同（参见 4.8 外生变量冲击）。注意，需要为首末指定时期之间的所有内生约束变量指定完整路径。如果未指定中间时期，则假定值为 0。也就是说，如果仅指定期间 1 和 3 的值，则期间 2 的值为 0。目前不存在不可控的中间时期。

然而，不同变量可能有不同数量的受控周期。此时的内生控制变量和 controlled_varexo 的声明顺序很重要：如果第二个内生变量的控制周期少于第一个，则不会为最后一个时期设置第二个 controlled_varexo。

存在 observation_trends 的情况下，这些变量的指定受控路径需要包括趋势分量。使用 *loglinear* 选项时，必须指定受控变量的对数。

Block:filter_initial_state;

卡尔曼滤波递归开始时内生状态的初始值。也就是说，如果卡尔曼滤波递归从时间$t = 1$为首个观测值开始，该模块提供时间0时刻的状态估计，即给定0时刻的信息$E_0(x_0)$。如果没有指定，初始条件假定为稳态（静态模型的无条件均值）。

模块以 end;结束。模块内的每行都应是这样的形式：

```
VARIABLE_NAME(INTEGER)=EXPRESSION;
```

EXPRESSION 是返回数值的任意有效表达式，可以包含参数值，允许在估计过程中遵循特定的关系。INTEGER 指变量出现的滞后期。根据 Dynare 的惯例，时期 1 是首期。向后追溯模拟开始前时期 0，然后是时期-1，以此类推。注意 filter_initial_state 模块不接受非状态变量。

示例

```
filter_initial_state;
k(0)=((1/bet-(1-del))/alp)^(1/(alp-1))*l_ss;
P(0)=2.5258;
m(0)=mst;
end;
```

Command:plot_conditional_forecast[VARIABLE_NAME...];

Command:plot_conditional_forecast(periods=INTEGER)[VARIABLE_NAME...];

绘制条件（直线）和无条件（虚线）预测，在 conditional_forecast 之后使用。

选项

periods=INTEGER：要绘制的周期数。默认值：等于 conditional_forecast 中的时期数。plot_conditional_forecast 中声明的周期数不能大于 conditional_forecast。

Command:bvar_forecast

计算样本外预测估计 BVAR 模型，使用 Minnesota 先验分布。了解更多相关信息命令参见与 Dynare 一同发布的 bvar-a-la-sims.pdf。

如果模型包含强非线性，或者考虑到一些完美预期冲击，预测和条件预测可以用扩展路径法计算。应当建立内生变量的冲击和/或约束路径的预测场景。第一步是使用函数 init_plan 初始化预测场景。

MATLAB/Octave command:HANDLE=init_plan(DATES);

为预测期创建新预测场景（表示为日期类，参见 6.1.2 日期类）。此函数得到新预测场景。预测场景可以包含一些外生变量的简单冲击。冲击用函数 basic_plan 描述。

MATLAB/Octave command:HANDLE=basic_plan(HANDLE,`VAR_NAME',`SHOCK_TYPE',DATES,MATLAB VECTOR OF DOUBLE|[DOUBLE|EXPR[DOUBLE|EXPR]]);

将第二个参数中引号之间指定的外生变量的冲击添加到预测场景中。冲击类型必须在引号之间的第三个参数中指定：“surprise”表示意外冲击，或“perfect_foresight”表示完全预期的冲击。第四个参数使用日期类指示冲击的时间段（参见 6.1.2 日期类）。最后一个参数是表示为 MATLAB 双精度向量的冲击路径。此函数得到更新后的预测场景。

预测场景还可以包含内生变量的受限路径。此时计算与约束路径兼容的相关外生变量的值。换句话说，执行有条件的预测。冲击用函数 flip_plan 来描述。

MATLAB/Octavecommand:HANDLE=flip_plan(HANDLE,`VAR_NAME',`VAR_NAME',`SHOCK_TYPE',DATES,MATLAB VECTOR OF DOUBLE|[DOUBLE|EXPR[DOUBLE|EXPR]]);

将第二个参数的引号之间指定的内生变量的约束路径添加到预测场景中。在引号之间的第三个参数中提供的相关外生变量视为内生变量，并且计算与内生变量上的约束路径兼容的值。约束路径上期望的性质必须在引号之间的第四个参数中指定：“surprise”表示意外路径，或“perfect_foresight”表示完全预期路径。第五个参数表示使用日期类（参见 6.1.2 日期类）约束内生变量路径的时间段。最后一个参数包含约束路径作为 MATLAB 向量。此函数得到更新后的预测场景。

一旦完全描述了预测场景，使用命令 det_cond_forecast 计算预测。

MATLAB/Octave command:DSERIES=det_cond_forecast(HANDLE[,DSERIES[,DATES]]);

使用给定预测场景（第一个参数）的扩展路径方法计算预测或条件预测。字符集类（参见 6.2 dseries 类）提供的内生变量和外生变量的过去值可以在第二个参数中指示。默认情况下，变量的过去值等于它们的稳态值。第三个参数提供预测的初始日期。默认情况下，预测将从 init_plan 命令中指定的首个日期开始。此函数得到一个包含内生变量和外生变量的历史和预测值的字符。

示例

```
% conditional forecast using extended path method
% with perfect foresight on r path

var y r;
varexo e u;
...
smoothed=dseries('smoothed_variables.csv');

fplan=init_plan(2013Q4:2029Q4);
fplan=flip_plan(fplan,'y','u','surprise',2013Q4:2014Q4,[1 1.
1 1.2 1.1]);
fplan=flip_plan(fplan,'r','e','perfect_foresight',2013Q4:2014
Q4,[2 1.9 1.9 1.9]);

dset_forecast=det_cond_forecast(fplan,smoothed);

plot(dset_forecast.{'y','u'});
plot(dset_forecast.{'r','e'});
```

Command:smoother2histval;

Command:smoother2histval(OPTIONS...);

构建初始条件（用于后续的模拟），这些初始条件是先验估计的平滑值。

更准确地说，使用 smoother 选项运行估计后，smoother2histval 将抽取平滑值（从 oo_.SmoothedVariables 抽取，有滞后外生变量可能从 oo_.SmoothedShocks 抽取），并将使用这些值构建初始条件（就好像通过 histval 手动输入）。

选项

period=INTEGER：后续模拟起点的周期编号，应该介于 1 和用于生成平滑值的观察数之间。默认值：最后一次观测值。

infile=FILENAME：从之前 Dynare 运行创建的_results.mat 文件加载平滑值。默认值：使用当前在全局工作区中的平滑值。

invars=(VARIABLE_NAME[VARIABLE_NAME...])：从平滑值中读取的变量列表。，可以包含状态内生变量，也可以包含具有滞后的外生变量。默认值：所有状态内生变量，以及所有具有滞后的外生变量。

outfile=FILENAME：初始条件写入文件。默认值：在当前工作空间中写入初始条

件以便模拟。

outvars=(VARIABLE_NAME[VARIABLE_NAME...])：给出初始条件的变量列表。此列表必须与提供给 invars 的列表具有相同的长度，并且两个列表之间存在一一映射。默认值：与选项 invars 相同的值。

使用案例

有三种可能的方式使用此命令：

- 单个文件中的所有内容：使用 smoother 运行估计，然后运行 smoother2histval（没有 infile 和 outfile 选项），然后运行随机模拟；
- 在两个文件中：第一个文件运行 smoother2histval，然后使用 outfile 选项运行 smoother2histval。第二个文件运行 histval_file 加载初始条件，并运行（确定性或随机性）模拟；
- 在两个文件中：第一个文件运行平滑器。第二个文件使用 infile 选项运行 smoother2histval，等价于第一个文件创建的_results.mat 文件，然后运行（确定性或随机性）模拟。

4.21 最优策略

Dynare 拥有为各种类型的目标计算最优策略的工具。可以使用 ramsey_model 解决承诺下的最佳策略，使用 discretionary_policy 解决求解自由裁量下的最优策略或使用 osr（也暗示承诺）解决最优简单规则。

Command:planner_objective MODEL_EXPRESSION;

声明政策制定者目标，与 ramsey_model 或 discretionary_policy 一起使用。

您需要给出一个时期的目标，而不是折现后整个时期的目标。折现因子由 ramsey_model 和 discretionary_policy 的 planner_discount 选项给出。目标函数只能包含当前的内生变量，不能包含外生变量。模型中定义适当的辅助变量可以轻松规避此约束。

使用 ramsey_model，您不仅限于二次目标函数，可以给出任意的非线性表达式。

使用 discretionary_policy，目标函数必须是二次的。

Command:evaluate_planner_objective;

在 Ramsey 或自由裁量权策略下，计算、显示并存储计划者目标函数的结果在 oo_.planner_objective_value。它将提供由计划者制定的内生变量和外生状态变量的初始值（即时期 0）的无条件福利和条件福利。确定性背景使用 initval 或 histval（取决于具体情况）设定各自的初始值。

随机性背景下，如果没有指定 histval 的初始状态值，它们的值将视为稳态值。因为条件福利是由计划者在第一个内生时期（时期 1）以最优政策为条件计算的，所以以时期 1 的信息集为条件。这个信息集包括从时期 0 继承的预定状态（通过内生和滞后外生状态的 histval 指定），以及外生冲击的时期 1 数值。后者是使用 shocks 模块的完美预期

语法指定的。

当前阶段，随机性背景不支持 pruning 选项。order>3 仅支持使用稳态拉格朗日乘子法计算条件福利。注意，order=2 时，输出结果是基于存储在 oo_.var 中方差的二阶精确近似值。

示例（随机性背景下）

```
var a ...;
varexo u;

model;
a=rho*a(-1)+u+u(-1);
...
end;

histval;
u(0)=1;
a(0)=-1;
end;

shocks;
var u; stderr 0.008;
var u;
periods 1;
values 1;
end;

evaluate_planner_objective;
```

MATLAB/Octave variable:oo_.planner_objective_value.unconditional

存储无条件福利值的标量。完美预期的背景下与长期福利保持一致，近似为最终模拟期的福利。

MATLAB/Octave variable:oo_.planner_objective_value.conditional

完美预期的背景下存储以指定初始条件和零初值拉格朗日乘子为条件的福利值标量。

随机性背景下有两个子字段：

MATLAB/Octave variable:oo_.planner_objective_value.conditional.steady_initial_multipliers

初始拉格朗日乘数关联计划者的目标时，存储计划者目标的值设置为稳态值（参见 *ramsey_policy*）。

MATLAB/Octave variable:oo_.planner_objective_value.conditional.zero_initial_multipliers

初始拉格朗日乘数关联计划者的目标时，存储计划者目标的值设置为 0。即假设计划者在实施 Ramsey 策略的第一阶段充分利用其能力让私人部门感到惊讶。这个值相当于计划者第一次执行最优策略，并承诺未来不再重新优化。

4.21.1 承诺下的最优策略（Ramsey）

Dynare 允许自动计算 Ramsey 计划者的最优政策选择，计划者考虑到指定的私营部门均衡条件并承诺未来的政策选择。这样做需要在 `model` 模块和 `planner_objective` 指定私营部门的均衡条件，以及可能的一些 `instruments` 促进计算。

警告：在永恒视角 Ramsey 计算场景下使用前向辅助变量时要小心。尽管它们似乎不影响私人部门的均衡，但可能会改变 Ramsey 计划者在第一时期要解决的问题。原因是计划者在日期t优化变量，并将时期 0 变量的值视为给定，因为它们是预先确定的。这组最初预先确定的变量将随着前向定义改变。因此，强烈建议用户改用 `model-local` 变量。

示例

以下是一个完全预测的示例，其中资本回报率的欧拉方程由下式给出：

```
1/C=beta*1/C(+1)*(R(+1)+(1-delta))
```

Ramsey 计划者在时期 1 的工作选择C_1和R_1，假设C_0已知。上述的等式可以等价于：

```
1/C=beta*1/C(+1)*(R_cap);
R_cap=R(+1)+(1-delta);
```

然而，由于完美预期，第一期 Ramsey 计划者的目标变为选择C_1和R_1，将C_0和R_0^{cap}视为给定。因此，由于第二行定义的前瞻性本质，第一期欧拉方程中相关的资本回报率不再是计划者的选择！

正确的做法是将 `R_cap` 定义为 `model-local` 变量：

```
1/C=beta*1/C(+1)*(R_cap);
#R_cap=R(+1)+(1-delta);
```

Command:ramsey_model(OPTIONS...);

计算了在私人经济均衡路径的约束下，使决策者目标函数最大化的一阶条件。必须使用 *planner_objective* 命令声明计划者目标。

此命令仅创建扩展模型，不执行任何计算。它需要跟随其他指令来实际执行所需的计算。例如，调用 `steady` 计算 Ramsey 经济的稳定状态，调用 `stoch_simul` 使用各种近似阶数执行基于扰动解的随机模拟，调用 `estimate` 旨在具有承诺的最优策略下估计模型以及完美预期模拟例程。

如何自动创建拉格朗日乘子的说明参见 4.6 辅助变量。

选项

此命令接受以下选项：

planner_discount=EXPRESSION：声明或重新分配中央计划者 optimal_policy_discount_factor 的折现因子。默认值：1.0。

planner_discount_latex_name=LATEX_NAME：设置 optimal_policy_discount_factor 参数的 LaTeX 名称。

instruments=(VARIABLE_NAME,...)：声明用于计算最优策略下稳态的工具变量。需要 steady_state_model 模块或_steadystate.m 文件，见下文。

稳态

Dynare 利用了拉格朗日乘子在最优策略下模型稳态方程中线性出现的事实。然而，只用 initval 中对内生变量的数字猜测值计算稳定状态非常困难。

如果用户提供稳态的解析解（steady_state_model 模块或_steadystate.m 文件）会极大地方便计算。此时有必要提供一个稳定状态的解决方案，它以最优策略问题中的工具价值为条件，并以选项 instruments 声明。此时用于寻找稳态的工具的初始值是用 initval 设置的。注意，使用 stable 命令或调用 resid 计算和显示稳态值必须在 ramsey_model 语句和 initval 模块之后。

注意，选择工具在一定程度上是一个解释问题，您可以选择从数学角度来看方便但不同于与论文分析引用的工具。典型例子是选择通货膨胀或名义利率作为工具。

Block:ramsey_constraints;

定义 Ramsey 问题中变量的约束。约束采用变量、不等式运算符（>或<）和常量形式。

示例

```
ramsey_constraints;
i>0;
end;
```

Command:ramsey_policy[VARIABLE_NAME...];

Command:ramsey_policy(OPTIONS...)[VARIABLE_NAME...];

形式上等价于调用序列：

```
ramsey_model;
stoch_simul;
evaluate_planner_objective;
```

计算政策的一阶近似值，该近似值在私人经济部门的均衡路径提供的约束下，在最优策略的承诺下，最大化决策者的目标函数。Ramsey 策略是通过逼近拉格朗日乘子稳态的扰动点附近的平衡系统计算的，即 Ramsey 计划者的行为就好像初始乘数在遥远的过去被设

置为 0，给它们时间来收敛到稳态值。因此，最优决策规则是围绕内生变量和拉格朗日乘子的这种稳态计算的。

注意，在 ramsey_policy 命令或 stoch_simul 命令之后的列表中的变量也可以包含乘子名称，但是对于大小写敏感（例 MULT_1）。例如，当 irf>0 时，Dynare 将显示各个乘数的 IRF。

必须使用 *planner_objective* 命令声明计划者目标。

选项

此命令接受 stoch_simul 的所有选项，此外：

planner_discount=EXPRESSION：参见 *planner_discount*.

instruments=(VARIABLE_NAME,...)：声明用于计算最优策略下稳态的工具变量。需要 steady_state_model 模块或_steadystate.m 文件。见下文。

输出

此命令生成 stoch_simul 的所有输出变量，要指定内生状态变量的初始值（拉格朗日乘子除外），参见前文。

稳态

参见 *Ramsey steady state*。

4.21.2 自由裁量权下的最优策略

Command:discretionary_policy[VARIABLE_NAME...];

Command:discretionary_policy(OPTIONS...)[VARIABLE_NAME...];

计算自由裁量权下最优策略的近似值。应用的算法本质是 Dennis（2007）描述的 LQ 求解器。应该确保目标是二次的，模型必须是线性的，或者在一阶上求解，并提供解析稳态。第一种情况还应该设置 model 模块的 linear 选项。

可以在 discretionary_policy 命令之后使用 *estimation* 命令，以便估计具有自由裁量权的最优策略模型，可用 *evaluate_planner_objective* 计算福利。

选项

接受与 ramsey_policy 相同的选项，此外：

discretionary_tol=NON-NEGATIVEDOUBLE：设置用于评估求解算法收敛性的容差级别。默认值：1e-7。

maxit=INTEGER：最大迭代次数。默认值：3000。

4.21.3 最优简单规则（OSR）

Command:osr[VARIABLE_NAME...];

Command:osr(OPTIONS...)[VARIABLE_NAME...];

为线性二次型问题计算最优简单策略规则：

$$\min_{\gamma} E\left(y_t' W y_t\right)$$

因此：

$$A_1 E_t y_{t+1} + A_2 y_t + A_3 y_{t-1} + C e_t = 0$$

其中：

- E：无条件期望运算符；
- γ：待优化参数。它们一定是矩阵A_1、A_2、A_3的元素，即在 params 命令中指定为参数，并输入到 model 模块；
- y：var 命令中指定的内生变量，（协）方差进入损失函数；
- e：varexo 命令中指定的外生随机冲击；
- W：加权矩阵；

线性二次方问题包括在模型的一阶条件所暗示的线性运动规律的约束下，选择一个模型参数子集，以最小化指定的内生变量子集的加权（协）方差。有几件事值得一提：第一，y表示选定的内生变量对稳态的偏离，即如果均值不为 0，则进入损失函数的变量会被自动去均值，从而最小化中心二阶矩；第二，osr 只解决指定的二次损失函数与模型平衡条件的一阶近似值相结合而产生的线性二次问题，原因是一阶状态空间表示用于计算无条件（协）方差。因此，osr 会自动选择 order=1；第三，因为目标涉及到最小化无条件二阶矩的加权和，二阶矩必须有限。特别是，不允许y有单位根。

优化简单规则的模型参数子集γ，必须用 osr_params 列出。二次目标函数的加权矩阵W在 optim_weights 模块指定。通过将权重附加到内生变量，进入目标函数的内生变量y的子集隐性指定的。使用 *opt_algo* 指定的数值优化器解决线性二次问题。

选项

osr 命令随后将运行 stoch_simul 并接受相同的选项，包括在命令后列出它们来约束内生变量，如 stoch_simul（参见 4.13 随机解和模拟）加上：

opt_algo=INTEGER：指定用于最小化目标函数的优化器。除了 5、6 和 10 之外，还可以使用与 mode_compute（参见 *mode_compute*）相同的求解器。

optim=(NAME,VALUE,...)：NAME 和 VALUE 的配对列表，可用于设置优化例程的选项。可用选项集取决于所选的优化例程（即选项 *opt_algo* 的值）。参见 *optim*。

maxit=INTEGER：确定在 opt_algo=4 中使用的最大迭代次数。此选项现已弃用，并将未来版本的 Dynare 删除。使用 optim 设置优化器特定的值。默认值：1000。

tolf=DOUBLE：基于 opt_algo=4 中使用的函数值的终值收敛标准。当证明超过 tolf 的方式改善函数值不成立时，停止迭代。此选项现已弃用，将在未来版本的 Dynare 删除。使用 optim 来设置优化器特定的值。默认值：e-7。

silent_optimizer：参见 *silent_optimizer*。

huge_number=DOUBLE：用有限数替换参数的无限边界的值。出于数字原因使用一些优化器（参见 *huge_number*）。用户需要确保最优参数不大于该值。默认值：1e7。

目标值存储在变量 oo_.osr.objective_function 中，在最优点的参数值存储在 oo_.osr.optim_params 中。更多细节参见下文。

运行 osr 后，进入简单规则的参数将设置为最优值，以便后续运行 stoch_simul 将以这些值执行。

Command:osr_params PARAMETER_NAME...;

声明要由 osr 优化的参数。

Block:optim_weights;

指定最优策略问题的二次目标。更准确地说，这个模块指定了权重矩阵W的非零元素，用于 osr 中目标函数的二次形式。

权重矩阵的对角线元素由以下形式的行给出：

```
VARIABLE_NAME EXPRESSION;
```

权重矩阵的非对角元素由以下形式的行给出：

```
VARIABLE_NAME,VARIABLE_NAME EXPRESSION;
```

示例

```
var y inflation r;
varexo y_ inf_;

parameters delta sigma alpha kappa gammarr gammax0 gammac0 ga
mma_y_ gamma_inf_;

delta= 0.44;
kappa= 0.18;
alpha= 0.48;
sigma=-0.06;

gammarr=0;
gammax0=0.2;
gammac0=1.5;
gamma_y_=8;
gamma_inf_=3;

model(linear);
y=delta*y(-1)+(1-delta)*y(+1)+sigma*(r-inflation(+1))+y_;
inflation=alpha*inflation(-1)+(1-alpha)*inflation(+1)+kappa*y
```

```
+inf_;
r=gammax0*y(-1)+gammac0*inflation(-1)+gamma_y_*y_+gamma_inf_*
inf_;
end;

shocks;
var y_; stderr 0.63;
var inf_; stderr 0.4;
end;

optim_weights;
inflation 1;
y 1;
y,inflation 0.5;
end;

osr_params gammax0 gammac0 gamma_y_ gamma_inf_;
osr y;
```

Block:osr_params_bounds;

在最优简单规则中声明参数的上下限。如果不指定，优化是无约束的。

每行有以下语法：

```
PARAMETER_NAME,LOWER_BOUND,UPPER_BOUND;
```

注意，使用这个模块需要约束优化器，即设置 *opt_algo* 到 1、2、5 或 9。

示例

```
osr_params_bounds;
gamma_inf_,0,2.5;
end;

osr(opt_algo=9) y;
```

MATLAB/Octave variable:oo_.osr.objective_function

执行 osr 命令后，包含最优策略下的目标值。

MATLAB/Octavevariable:oo_.osr.optim_params

执行 osr 命令后，包含在最优点的参数值，存储在 oo_.osr.optim_params.PARAMETER_NAME 形式的字段中。

MATLAB/Octave variable:M_.osr.param_names

执行 osr 命令后，包含参数的名称。

MATLAB/Octave variable:M_.osr.param_indices

执行 osr 命令后，包含 M.params 中 OSR 参数的索引。

MATLAB/Octave variable:M_.osr.param_bounds

执行 osr 命令后，2×OSR 参数矩阵的数量分别包含第一、二列参数的下限和上限。

MATLAB/Octave variable:M_.osr.variable_weights

执行 osr 命令后，该稀疏矩阵包含与目标函数中的变量关联的加权矩阵。

MATLAB/Octave variable:M_.osr.variable_indices

执行 osr 命令后，该向量包含在 M_.endo_names 中进入目标函数的变量索引。

4.22 敏感度和识别分析

Dynare 为全部灵敏度分析（GSA）工具箱（由欧盟委员会联合研究中心（JRC）开发）提供了接口，该工具箱现已成为正式 Dynare 分布的一部分。GSA 工具箱可回答以下问题：

1. 保证 DSGE 模型稳定性和确定性的结构系数域是什么？
2. 哪些参数主要驱动 GDP 的拟合？哪些是通货膨胀的拟合？一个观察序列的最优拟合与另一个观察序列之间是否存在冲突？
3. 如何直接（尽管近似）地表示结构参数与理性预期模型的简化形式之间的关系？

关于这些方法及其应用的讨论记载于 Ratto（2008）。

为了正常工作，关于工具箱的上一版本，GSA 工具箱不再需要设置 Dynare 估计环境。

4.22.1 执行敏感性分析

Command:dynare_sensitivity;

Command:dynare_sensitivity(OPTIONS...);

触发对 DSGE 模型的敏感性分析。

选项

Nsam=INTEGER：蒙特卡洛样本的大小。默认值：2048。

ilptau=INTEGER：如果等于 1，则使用LP_τ拟蒙特卡洛。如果等于 0，使用 LHS 蒙特卡洛。默认值：1。

pprior=INTEGER：如果等于 1，则从先前的分布中抽取样本。如果为 0，则从多元正态$N(\bar{\theta}, \Sigma)$的样本，其中$\bar{\theta}$为后验众数，$\Sigma = H^{-1}$，H为该众数的海塞矩阵。默认值：1。

prior_range=INTEGER：如果等于 1，则按先验范围均匀取样。如果等于 0，则从先前的分布中抽取样本。默认值：1。

morris=INTEGER：如果等于 0，ANOVA 映射（I 类错误）等于 1 时，筛选分析（II 类错误）等于 2 时，解析导数（II 类错误，只有在标识=1 时有效）。默认值：当 identification=1 为 1，否则为 0。

morris_nliv=INTEGER：Morris 设计中的层次数。默认值：6。

morris_ntra=INTEGER：Morris 设计中的数字轨迹。默认值：20。

ppost=INTEGER：如果等于 1，则使用 Metropolis 采样算法后验样本。如果等于 0，则不用后验样本。默认值：0（注：这优先于任何其他抽样选项）。

neighborhood_width=DOUBLE：当 ppry=0 和 ppost=0 时，允许在 mode_file 中指定的值的周围对参数进行采样，范围为 xparam1±|xparam1×neighborhood_width|。默认值：0。

稳定性映射选项

stab=INTEGER：如果等于 1，则执行稳定映射。如果等于 0，则不执行稳定映射。默认值：1。

load_stab=INTEGER：如果等于 1，则加载先前创建的示例。如果等于 0，则生成新的示例。默认值：0。

alpha2_stab=DOUBLE：滤波样本中相关系数ρ的临界值：绘制$|\rho|$>alpha2_stab 参数配对图。默认值：0。

pvalue_ks=DOUBLE：重要 Kolmogorov-Smirnov 检验的门槛值*pvalue*（即*pvalue*<pvalue_ks 的绘图参数）。默认值：0.001。

pvalue_corr=DOUBLE：滤波样本中显著相关的门槛值*pvalue*（即当*pvalue*<pvalue_corr 时绘制双变量样本）。默认值：1e-5。

简化表单映射选项

redform=INTEGER：如果等于 1，则准备简化矩阵的蒙特卡洛样本。如果等于 0，不准备简化矩阵的蒙特卡洛样本。默认值：0。

load_redform=INTEGER：如果等于 1，加载先前估计的映射。如果等于 0，则估计简化式模型的映射。默认值：0。

logtrans_redform=INTEGER：如果等于 1，则使用日志转换项。如果等于 0，则使用原始条目。默认值：0。

threshold_redform=[DOUBLE DOUBLE]：分析简约形式系数滤波的蒙特卡洛项的范围。第一个数是下界，第二个数是上界。空向量指示不会过滤这些条目。默认值：空。

ksstat_redform=DOUBLE：滤波简化式条目时 Smirnov 统计信息d的临界值。默认值：0.001

alpha2_redform=DOUBLE：滤波简化式条目时相关系数临界值ρ。默认值：1e-5。

namendo=(VARIABLE_NAME...)：内生变量列表。“:”表示所有内生变量。默认值：空。

namlagendo=(VARIABLE_NAME...)：滞后内生变量列表。“:”表示所有滞后的内生变量。分析条目[namendo×namlagoro]。默认值：空。

namexo=(VARIABLE_NAME...)：外生变量列表。“:”表示所有外生变量。分析条目[namendo×namexo]。默认值：空。

RMSE 选项

rmse=INTEGER：如果等于 1，则执行 RMSE 分析。如果等于 0，则不执行 RMSE 分析。默认值：0。

load_rmse=INTEGER：如果等于 1 时，加载先前的 RMSE 分析。如果等于 0，则执行新的 RMSE 分析。默认值：0。

lik_only=INTEGER：如果等于 1，则只计算似然和后验。如果等于 0，则计算所有观测序列的 RMSE。默认值：0。

var_rmse=(VARIABLE_NAME...)：待查的观察序列清单。“:”表示所有观察到的变量。默认值：varobs。

pfilt_rmse=DOUBLE：RMSE 滤波门槛值。默认值：0.1。

istart_rmse=INTEGER：开始计算 RMSE 的值(使用 2 避免大的初始错误)。默认值：presample+1。

alpha_rmse=DOUBLE：Smirnov 统计数据d的临界值：d>alpha_rmse 的绘图参数。默认值：0.001。

alpha2_rmse=DOUBLE：相关系数ρ的临界值：带有绘制$|\rho|$>alpha2_rmse 的参数配对图。默认值：1e-5。

datafile=FILENAME：参见 *datafile*。

nobs=INTEGER

nobs=[INTEGER1:INTEGER2]：参见 *nobs*。

first_obs=INTEGER：参见 *first_obs*。

prefilter=INTEGER：参见 *prefilter*。

presample=INTEGER：参见 *presample*。

nograph：参见 *nograph*。

nodisplay：参见 *nodisplay*。

graph_format=FORMAT graph_format=(FORMAT,FORMAT...)：参见 *graph_format*。

conf_sig=DOUBLE：参见 *conf_sig*。

loglinear：参见 *loglinear*。

mode_file=FILENAME：参见 *mode_file*。

kalman_algo=INTEGER：参见 *kalman_algo*。

识别分析选项

identification=INTEGER：如果等于 1，则执行识别分析（强制 redform=0 和

morris=1)，如果等于 0，则不执行。默认值：0。

morris=INTEGER：参见 *morris*。

morris_nliv=INTEGER：参见 *morris_nliv*。

morris_ntra=INTEGER：参见 *morris_ntra*。

load_ident_files=INTEGER：先前进行的识别分析。默认值：0。

useautocorr=INTEGER：用自相关矩阵代替矩的自协方差矩阵执行辨识分析。默认值：0。

ar=INTEGER：识别分析中矩的最大滞后数。默认值：1。

diffuse_filter=INTEGER：参见 *diffuse_filter*。

4.22.2 IRF/矩校准

irf_calibration 和 moment_calibration 模块允许在模型上施加 IRF 和矩的隐式“内生”先验。内部工作的方式是舍弃任何与这些模块中提供的“校准”不一致的参数抽样，即分配先验密度 0。在 dynare_sensitive 的环境中，这些约束允许追踪出哪些参数在驱动模型满足或违反给定的约束。

IRF 和矩校准可以定义在 irf_calibration 和 moment_calibration 模块：

Block:irf_calibration;

Block:irf_calibration(OPTIONS...);

允许定义 IRF 校准标准，并由 end;终止。要设置 IRF 符号约束，使用以下语法：

```
VARIABLE_NAME(INTEGER),EXOGENOUS_NAME,-;
VARIABLE_NAME(INTEGER:INTEGER),EXOGENOUS_NAME,+;
```

若要设置具有特定区间的 IRF 约束，使用以下语法：

```
VARIABLE_NAME(INTEGER),EXOGENOUS_NAME,[EXPRESSION,EXPRESSIO
N];
VARIABLE_NAME(INTEGER:INTEGER),EXOGENOUS_NAME,[EXPRESSION,EXP
RESSION];
```

当使用(INTEGER:INTEGER)时，该约束认为是由逻辑 OR 实现的。约束列表必须始终满足逻辑 AND。

选项

relative_irf：参见 *relative_irf*。

示例

```
irf_calibration;
y(1:4),e_ys,[-50,50];//[first year response with logical OR]
@#for ilag in 21:40
R_obs(@{ilag}),e_ys,[0,6];//[response from 5th to 10th years
```

```
with logical AND]
@#endfor
end;
```

Block:moment_calibration;

Block:moment_calibration(OPTIONS...);

许定义力矩校准标准，并由 end;终止，包含表单的行：

```
VARIABLE_NAME1,VARIABLE_NAME2(+/-INTEGER),[EXPRESSION,EXPRESS
ION];
VARIABLE_NAME1,VARIABLE_NAME2(+/-INTEGER),+/-;
VARIABLE_NAME1,VARIABLE_NAME2(+/-(INTEGER:INTEGER)),[EXPRESSI
ON,EXPRESSION];
VARIABLE_NAME1,VARIABLE_NAME2((-INTEGER:+INTEGER)),[EXPRESSIO
N,EXPRESSION];
```

当使用(INTEGER:INTEGER)时，该约束认为是由逻辑 OR 实现的。约束列表必须始终满足逻辑 AND。

示例

```
moment_calibration;
y_obs,y_obs,[0.5,1.5];//[unconditional variance]
y_obs,y_obs(-(1:4)),+;//[sign restriction for first year autocorr
elation with logical OR]
@#for ilag in -2:2
y_obs,R_obs(@{ilag}),-;//[-2:2 cross correlation with logical AN
D]
@#endfor
@#for ilag in -4:4
y_obs,pie_obs(@{ilag}),-;//[-4_4 cross correlation with logical A
ND]
@#endfor
end;
```

4.22.3 执行识别分析

Command:identification;

Command:identification(OPTIONS...);

该命令触发：

1. 基于理论识别分析：

- Iskrev（2010）的矩；
- Qu 和 Tkachenko（2012）的谱密度；
- Komunjer 和 Ng（2011）的最小系统；
- Ratto 和 Iskrev（2011）的简化形式解和线性理性期望模型。

注意，对于阶数 2 和 3，所有识别检查均基于 Mutschler（2015）中的修剪状态空间系统。也就是说，理论矩和谱是根据修剪后的 ABCD 系统计算的，而最小系统标准基于一阶系统，但通过二阶或三阶的理论（修剪）平均值增强。

2. 基于（理论或模拟）矩信息矩阵曲率的识别强度分析，如 Ratto 和 Iskrev（2011）；

3. 基于零空间和多相关系数的参数检查，以确定涉及哪些（组合）参数。

选项

order=1|2|3：近似顺序。在二阶和三阶识别是基于修剪过的状态空间系统。注意，其他函数设置的顺序不会覆盖默认值。默认值：1。

parameter_set=OPTION：可能值参见 *parameter_set*。默认值：prior_mean。

prior_mc=INTEGER：蒙特卡洛样本的大小。默认值：1。

prior_range=INTEGER：在先前规范暗示的范围内触发统一采样（当 prior_mc> 1）。默认值：0。

advanced=INTEGER：如果设置为 1，则显示更详细的分析，包括对线性化理性期望模型的分析以及关联的简化形式解。进一步对最能再现每个单一参数行为的参数组实行暴力搜索。搜索到的组的最大维度由 max_dim_cova_group 触发。默认值：0。

max_dim_cova_group=INTEGER：暴力搜索（advanced=1 时执行）中，此选项设置参数组的最大维度，这些参数组最能再现每个单一模型参数的行为。默认值：2。

gsa_sample_file=INTEGER|FILENAME：如果等于 0，则不使用示例文件。如果等于 1，则触发 gsa 先验采样。如果等于 2，则触发 gsa 蒙特卡洛抽样（即加载对应于 dynare_sensitivity 选项中 pprior=0 和 ppost=0 的样本）。如果等于 FILENAME，则使用提供的特定用户定义示例文件的路径。默认值：0。

diffuse_filter：处理非平稳情况。参见 *diffuse_filter*。

数值选项

analytic_derivation_mode=INTEGER：以解析或数值形式计算导数的不同方法。可能的值是：

0：计算解析导数的高效 sylvester 方程方法；

1：计算分析导数的 Kronecker 积法（仅在 order=1 时）；

-1：计算所有识别雅可比行列式的数值两侧有限差分法（数值容差水平等于 options_.dynatol.x）；

-2：数值两侧有限差分法，以数值方式计算稳态和动态模型的导数，然后解析计算识

别雅可比行列式（数值公差水平等于 options_.dynatol.x）。

默认值：0。

normalize_jacobians=INTEGER：如果设置为 1，通过按绝对值中的最大元素重新缩放每行来规范化雅可比行列式。通过转换为相关类型矩阵来标准化 Gram（或 Hessian 类型）矩阵。默认值：1。

tol_rank=DOUBLE：秩计算的容差水平。默认值：1.e-10。

tol_deriv=DOUBLE：在雅可比行列式中选择非零列的容差水平。默认值：1.e-8。

tol_sv=DOUBLE：选择非零奇异值的容差水平。默认值：1.e-3。

schur_vec_tol=DOUBLE：参见 *schur_vec_tol*。

识别强度选项

no_identification_strength：禁用基于样本信息矩阵的识别强度分析计算。

periods=INTEGER：解析海塞矩阵不可用（即具有缺失值或扩散卡尔曼滤波器或单变量卡尔曼滤波器）时，这会触发随机模拟的长度以计算模拟矩不确定性。默认值：300。

replic=INTEGER：当解析海塞矩阵不可用时，这会触发副本数量以计算模拟矩不确定性。默认值：100。

矩选项

no_identification_moments：禁用基于 Iskrev（2010）的 J（即前两个矩的导数）的识别检查计算。

ar=INTEGER：Iskrev（2010）的 J 标准计算自协方差/自相关（理论矩）的滞后数。默认值：1。

useautocorr=INTEGER：如果等于 1，则计算自相关的导数。如果等于 0，则计算自协方差的导数。默认值：0。

频谱选项

no_identification_spectrum：禁用基于 Qu 和 Tkachenko（2012）的 G 识别检查计算，即一阶矩导数的 Gram 矩阵加上谱密度导数的外积。

grid_nbr=INTEGER：[-pi;pi]的网格点数近似积分，计算 Qu 和 Tkachenko（2012）的 G 标准。默认值：5000。

最小状态空间系统选项

no_identification_minimal：禁用基于 Komunjer 和 Ng（2011）的 D 识别检查计算，即最小状态空间系统和观测等效谱密度变换。

Misc 选项

nograph：参见 *nograph*。

nodisplay：参见 *nodisplay*。

graph_format=FORMAT graph_format=(FORMAT,FORMAT...)：参见 *graph_*

format。

tex：参见 *tex*。

调试选项

load_ident_files=INTEGER：如果等于 1，则允许 Dynare 加载先前计算的分析。默认值：0。

lik_init=INTEGER：参见 *lik_init*。

kalman_algo=INTEGER：参见 *kalman_algo*。

no_identification_reducedform：禁用基于稳态和简化形式解的识别检查计算。

checks_via_subsets=INTEGER：如果等于 1，以暴力方式查找有问题的参数：它计算所有可能参数组合的雅可比行列式。如果秩条件未满足，这些参数集将标记为不可识别。搜索到的组的最大维度由 max_dim_subsets_groups 触发。默认值：0。

max_dim_subsets_groups=INTEGER：设置执行上述暴力搜索的参数组的最大维度。默认值：4。

4.22.4 分析和输出文件类型

敏感性分析工具箱包括若干类型的分析，结果保存在本地<mod_file>/gsa 中，其中<mod_file>.mod 是 Dynare 模型文件的名称。

4.22.4.1 抽样

生成以下二进制文件：

- <mod_file>_prior.mat：存储先验采样分析信息，即 pprior=1 和 ppost=0；
- <mod_file>_mc.mat：存储从多元正态采样分析信息，即 pprior=0 和 ppost=0；
- <mod_file>_post.mat：存储使用 Metropolis 采样算法后验样本分析信息，即 ppost=1。

4.22.4.2 稳定映射

生成的图形文件格式为<mod_file>_prior_*.fig 并存储先前蒙特卡洛样本的稳定性映射结果：

- <mod_file>_prior_stable.fig：Smirnov 检验和相关分析的图片，其中满足 Blanchard-Kahn 条件的样本累积分布函数（蓝色）与其余样本的累积分布函数（红色），即不稳定性或不确定性或者无法找到解决方案（例如，求解器无法求解稳态）；
- <mod_file>_prior_indeterm.fig：Smirnov 检验和相关分析的图片，产生不确定性样本的累积分布函数（红色）与其余样本累积分布函数（蓝色）对应；
- <mod_file>_prior_unstable.fig：Smirnov 检验和相关分析的图片，产生

膨胀根样本的累积分布函数（红色）与其余样本累积分布函数（蓝色）对应；

- <mod_file>_prior_wrong.fig：Smirnov 检验和相关分析的图片，其中无法找到解决方案样本的累积分布函数（例如，求解器无法求解稳态——红色）与其余样本累积分布函数（蓝色）对应；
- <mod_file>_prior_calib.fig：Smirnov 检验和相关分析的图片，通过对比 IRF/矩约束匹配的样本的累积分布函数（蓝色）和 IRF/矩约束不匹配的累积分布函数（红色），分割满足 Blanchard-Kahn 条件的样本。

类似惯例适用于使用多元正态的样本时获得的<mod_file>_mc_*.fig 文件。

4.22.4.3 IRF/矩约束

生成以下二进制文件：

- <mod_file>_prior_restrictions.mat：存储从先验范围采样的 IRF/矩约束分析的信息，即 pprior=1 和 ppost=0；
- <mod_file>_mc_restrictions.mat：存储从多元正态采样执行的 IRF/矩约束分析的信息，即 pprior=0 和 ppost=0；
- <mod_file>_post_restrictions.mat：存储使用 Metropolis 采样算法后验样本执行的 IRF/矩约束分析的信息，即 ppost=1。

生成的图形文件的格式为<mod_file>_prior_irf_calib_*.fig 和<mod_file>_prior_moment_calib_*.fig 并存储先前蒙特卡洛样本的映射约束结果：

- <mod_file>_prior_irf_calib_<ENDO_NAME>_vs_<EXO_NAME>_<PERIOD>.fig：Smirnov 检验和相关分析的图片，通过对比期间<PERIOD>个别 IRF 约束<ENDO_NAME>和<EXO_NAME>匹配样本的累积分布函数（蓝色）与 IRF 约束不匹配的累积分布函数（红色），分割满足 Blanchard-Kahn 条件的样本；
- <mod_file>_prior_irf_calib_<ENDO_NAME>_vs_<EXO_NAME>_ALL.fig：Smirnov 检验和相关性分析的图片，通过对比同组<ENDO_NAME>和<EXO_NAME>的所有个别 IRF 约束匹配样本的累积分布函数（蓝色）与 IRF 约束不匹配的累积分布函数（红色），分割满足 Blanchard-Kahn 条件的样本；
- <mod_file>_prior_irf_restrictions.fig：与先验样本的实际蒙特卡洛实现相比，绘制 IRF 约束的视觉信息；
- <mod_file>_prior_moment_calib_<ENDO_NAME1>_vs_<ENDO_NAME2>_<LAG>.fig：Smirnov 检验和相关性分析的图片，通过对比在滞后<LAG>处个别自相关函数或互相关函数的矩约束<ENDO_NAME1>和<ENDO_NAME2>匹配样本的累积分布函数（蓝色）和 IRF 约束不匹配的累积分布函数（红色），分割满足 Blanchard-Kahn 条件的样本；
- <mod_file>_prior_moment_calib_<ENDO_NAME>_vs_<EXO_NAME>_ALL.

fig：Smirnov 检验和相关性分析的图片，通过对比同组<ENDO_NAME>和<EXO_NAME>的所有个别自相关函数或互相关函数的矩约束匹配样本的累积分布函数（蓝色）与 IRF 约束不匹配的累积分布函数（红色），分割满足 Blanchard-Kahn 条件的样本；

- <mod_file>_prior_moment_restrictions.fig：与先验样本的实际蒙特卡洛实现相比，绘制矩约束的视觉信息。

类似的惯例适用于<mod_file>_mc_*.fig 和<mod_file>_post_*.fig 文件，当使用来自多元正态或后验的样本时获得。

4.22.4.4 简化式映射

没有设置选项 threshold_redform，或是空时（默认），该分析对模型的简化形式一阶解冲击矩阵的转移矩阵中的选定条目，估计一个多变量平滑样条 ANOVA 模型（“映射”）。这个映射是用先验样本或 neighbour_width 的 MC 样本完成。除非用 MC 样本设置 neighbour_width，否则简化形式解的映射强制使用先验范围或先验分布的样本，即 prior=1 和 ppost=0。使用 250 个样本优化平滑参数，1000 个样本计算拟合。其余的样本则用于样本外的验证。也可以用新的蒙特卡洛样本加载先前估计的映射，观察新的蒙特卡洛样本的预测。

制作如下合成图：

- <mod_file>_redform_<endo name>_vs_lags_*.fig：显示了驱动所选内生变量（namendo）与滞后内生变量（namlagendo）的简化形式系数的十个最重要参数的敏感性指数条形图。后缀 log 表示日志转换条目的结果；
- <mod_file>_redform_<endo name>_vs_shocks_*.fig：显示了驱动所选内生变量（namendo）与外生变量（namexo）的简化形式系数的十个最重要参数的敏感性指数条形图；后缀 log 表示日志转换条目的结果；
- <mod_file>_redform_gsa(_log).fig：显示每个参数的所有敏感度指数条形图，允许人们注意到对任何简化形式系数具有较小影响的参数。

分析的详细结果，先验样本显示在子文件夹<mod_file>/gsa/redform_prior，带有选项 neighbor_width 的 MC 样本在<mod_file>/gsa/redform_mc，其中参数θ和简化形式系数（以下表示为y）之间单个函数关系的具体估计结果存储在单独的目录，命名为：

- <namendo>_vs_<namlagendo>：转换矩阵的条目；
- <namendo>_vs_<namexo>：冲击矩阵的条目。

以下文件存储在每个目录中（本文坚持使用先前的示例，但类似的惯例用于 MC 样本）：

- <mod_file>_prior_<namendo>_vs_<namexo>.fig：冲击矩阵单个条目的 MC 样本的直方图和 CDF 图，ANOVA 模型的样本内和样本外拟合；

- <mod_file>_prior_<namendo>_vs_<namexo>_map_SE.fig：对于冲击矩阵的条目，显示了每个深度参数θ_i的估计一阶 ANOVA 项$y = f(\theta_i)$的图形；
- <mod_file>_prior_<namendo>_vs_<namlagendo>.fig：转移矩阵的单个条目的 MC 样本的直方图和 CDF 图，ANOVA 模型的样本内和样本外拟合；
- <mod_file>_prior_<namendo>_vs_<namlagendo>_map_SE.fig：对于转移矩阵的条目，显示了每个深度参数θ_i的估计一阶 ANOVA 项$y = f(\theta_i)$的图形；
- <mod_file>_prior_<namendo>_vs_<namexo>_map.mat、<mod_file>_<namendo>_vs_<namlagendo>_map.mat：这些文件在估计中存储信息。

设置选项 logtrans_redform 时，使用每个y的对数变换执行 ANOVA 估计。然后将 ANOVA 映射转换回原始比例，以允许和基线估计可比。此日志转换案例的图形存储在以_log 后缀表示的文件中的同一文件夹中。

设置选项 threshold_redform 时，通过蒙特卡洛滤波执行分析，通过显示在 threshold_redform 指定的范围内驱动单个条目 y 的参数。如果未找到条目（或所有条目都在范围内），则 MCF 算法将忽略 threshold_redform 指定的范围，并执行将 y 的 MC 样本拆分为十分位数的分析。设置 threshold_redform=[-inf inf]会为所有 y 触发此方法。结果存储在名为<mod_file>/gsa/redform_prior 的子目录中：

- <mod_file>_prior_<namendo>_vs_<namlagendo>_threshold：转移矩阵的条目；
- <mod_file>_prior_<namendo>_vs_<namexo>_threshold：冲击矩阵的条目。

保存的文件名为：

- <mod_file>_prior_<namendo>_vs_<namexo>_threshold.fig、<mod_file>_<namendo>_vs_<namlagendo>_threshold.fig：图形输出；
- <mod_file>_prior_<namendo>_vs_<namexo>_threshold.mat、<mod_file>_<namendo>_vs_<namlagendo>_threshold.mat：分析信息。

4.22.4.5 RMSE

可以使用不同类型的采样选项执行 RMSE 分析：

- 当 pprior=1 和 ppost=0 时，工具箱分析通过从其先验分布（或先验范围）中采样参数获得的蒙特卡洛样本的 RMSE：此分析提供了一些提示，即在全面估算之前哪些参数驱动了哪个观察序列的拟合；
- 当 pprior=0 和 ppost=0 时，工具箱分析多元正态蒙特卡洛样本 RMSE，协方差矩阵基于最优点的逆海塞矩阵：完成最大似然估计（即没有贝叶斯估计）时，此分析很有用；
- 当 ppost=1 时，工具箱分析通过 Dynare 的 Metropolis 采样算法程序获得后验样

本的 RMSE。

使用案例 2 和案例 3 需要事先估计步骤。为方便估计后的敏感性分析，dynare_sensivity 命令还允许指示 estimation command 的以下这些选项：

- datefile；
- nobs；
- first_obs；
- prefilter；
- presample；
- nograph；
- nodisplay；
- graph_format；
- conf_sig；
- loglinear；
- mode_file。

生成 RMSE 分析的二进制文件：

- <mod_file>_prior_*.mat：存储了先验蒙特卡洛样本的过滤和平滑变量，在进行 RMSE 分析时生成（pprior=1 和 ppost=0）；
- <mode_file>_mc_*.mat：存储了多元正态蒙特卡洛样本的过滤和平滑变量，在进行 RMSE 分析时生成（pprior=0 和 ppost=0）。

图文件<mod_file>_rmse_*.fig 存储 RMSE 分析的结果。

- <mod_file>_rmse_prior*.fig：使用先前的蒙特卡洛样本保存分析结果；
- <mod_file>_rmse_mc*.fig：使用多元正态蒙特卡洛样本保存分析结果；
- <mod_file>_rmse_post*.fig：使用 Metropolis 采样算法后验样本保存分析结果。

保存了以下类型的图形（显示了先验样本修复想法，但相同惯例用于多元正态和后验）：

- <mod_file>_rmse_prior_params_*.fig：对于每个参数，绘制对应每个观察序列的最优 10%RMSE 的累积分布函数（仅低于显著性阈值 alpha_rmse 的累积分布函数）；
- <mod_file>_rmse_prior_<var_obs>_*.fig：如果一个参数显著影响 var_obs 的拟合，则绘制了同一参数下与其他观测指标的所有可能的权衡结果；
- <mod_file>_rmse_prior_<var_obs>_map.fig：绘制显著地推动观察序列 var_obs 的拟合参数的 MCF 分析；
- <mod_file>_rmse_prior_lnlik*.fig：对于每个观察序列，蓝色绘制对应最优 10%RMSE 的对数似然累积分布函数，红色绘制其余样本的对数似然累积分

布函数，黑色绘制完整样本的对数似然累积分布函数。允许看到一些特殊行为；

- <mod_file>_rmse_prior_lnpost*.fig：对于每个观察序列，蓝色绘制对应最优 10%RMSE 的对数后验累积分布函数，红色绘制其余样本的对数似然累积分布函数，黑色绘制完整样本的对数似然累积分布函数。允许看到一些特殊行为；
- <mod_file>_rmse_prior_lnprior*.fig：对于每个观察序列，蓝色绘制对应最优 10%RMSE 的对数先验累积分布函数，红色绘制其余样本的对数似然累积分布函数，黑色绘制完整样本的对数似然累积分布函数。允许看到一些特殊行为；
- <mod_file>_rmse_prior_lik.fig：lik_only=1 时，显示了滤波最优 10%对数似然值的 MCF 检验；
- <mod_file>_rmse_prior_post.fig：当 lik_only=1 时，显示了滤波最优 10%对数后验值的 MCF 检验。

4.22.4.6 筛选分析

筛选分析不需要在采样选项中列出的任何额外选项。工具箱执行需要的所有分析并显示结果。使用 Morris 抽样设计的筛选分析结果存储在子文件夹<mod_file>/gsa/screen 中。数据文件<mod_file>_prior 存储所有分析信息（Morris 样本、简化式系数等）。

筛选分析仅涉及简化式系数，保存了类似于用蒙特卡洛样本执行简化式分析的合成条形图：

- <mod_file>_redform_<endo name>_vs_lags_*.fig：显示了驱动选定的内生变量（namendo）与滞后内生变量（namlagendo）的简化式系数的十个最重要参数的基本效应检验条形图；
- <mod_file>_redform_<endo name>_vs_shocks_*.fig：显示了驱动选定的内生变量（namendo）与外生变量（namexo）的简化式系数的十个最重要参数的基本效应检验条形图；
- <mod_file>_redform_screen.fig：显示每个参数的所有基本效果检验条形图：允许识别对任何简化式系数具有轻微影响的参数。

4.22.4.7 识别分析

设置选项 identification=1，就会执行基于理论矩的识别分析。提供敏感性图，可以推断出哪些参数最有可能不易识别。

正确运行所有识别程序的先决条件是 Dynare 模型文件中的关键词识别。这个关键词触发了模型对待估参数和冲击的解析导数的计算。这是 morris=2 选项的要求，该选项实现了 Iskrev（2010）的识别分析。

例如放置：

```
identification;
dynare_sensitivity(identification=1,morris=2);
```

在 Dynare 模型文件中，使用 Iskrev（2010）中的解析导数触发识别分析，并结合可接受区域的映射。

也可以通过单个命令触发具有导数的识别分析：

```
identification;
```

这不会为模型做可接受区域的映射，而是使用 Dynare 的标准随机采样器。此外，仅使用 identification；添加了两个额外的识别检查：即基于谱密度的 Qu 和 Tkachenko（2012）和基于最小状态空间系统的 Komunjer 和 Ng（2011）。它完全抵消了敏感性分析工具箱的任意使用。

4.23 马尔科夫转换 SBVAR

给定变量列表、观察变量和数据文件，根据 Sims、Wagoner 和 Zha（2008），Dynare 可解决马尔科夫转换 SBVAR 模型。[12]完成此操作后，可以创建预测并计算模型的边际数据密度、状态概率、脉冲响应函数和方差分解。

这些命令已经模块化，允许在<mod_file>.mod 文件中多次调用同一命令。默认使用<mod_file>标记程序使用（产生）的输入（输出）文件。因此，要在<mod_file>.mod 文件中多次调用任何命令，必须使用下面描述的*_tag 选项。

Command:markov_switching(OPTIONS...);

声明马尔科夫转换 SBVAR 模型的马尔科夫状态变量信息。

选项

chain=INTEGER：考虑的马尔科夫链。默认值：none

number_of_regimes=INTEGER：指定马尔科夫链状态的总数。必选项。

duration=DOUBLE|[ROW VECTOR OF DOUBLES]：状态的持续时间。必选项。当传递一个标量实数时，它指定了该链中所有状态的平均持续时间。当传递大小等于 number_of_regimes 的向量时，它指定了该链中相关状态（1:number_of_regimes）的平均持续时间。可以通过 restrictions 选项指定吸收状态。

restrictions=[[ROW VECTOR OF 3 DOUBLES],[ROW VECTOR OF 3 DOUBLES],...]：提供链的状态转换矩阵的约束。向量参数采用以下形式的三个输入：[current_period_regime,next_period_regime,transition_probability]。

前两项是正整数，第三项是集合[0,1]中的非负实数。如果为某个状态的每个转换指定了约束，则概率之和必须为 1。否则，如果没有为给定状态的每个转换提供约束，则所提供的转换概率之和必须小于 1。无论滞后次数如何，都为时间 t 的参数指定了约束，因为 t 处参数的转移概率等于 t-1 处参数的转移概率。

为了估计 MS-DSGE 模型，[13]允许有以下选项：

12 如果您想将论文与此处的描述一致，注意A是A^0，F是A^+。

13 示例可以在 https://git.dynare.org/Dynare/dynare/blob/master/tests/ms-dsge/test_ms_dsge.mod 找到。

parameters=[LIST OF PARAMETERS]：明确了马尔科夫链控制的变量。

number_of_lags=DOUBLE：提供此链中每个状态下每个参数所能承受的滞后数。

示例

```
markov_switching(chain=1,duration=2.5,restrictions=[[1,3,0],
[3,1,0]]);
```

指定一个马尔科夫转换 BVAR，第一条链具有 3 个持续时间均为 2.5 个周期的状态。从状态 1 直接进入状态 3 以及反过来的概率为 0。

示例

```
markov_switching(chain=2,number_of_regimes=3,duration=[0.5,
2.5,2.5],parameter=[alpha,rho],number_of_lags=2,restrictions
=[[1,3,0],[3,3,1]]);
```

指定一个马尔可夫转换 DSGE 模型，其中第二条链具有 3 个持续时间分别为 0.5、2.5 和 2.5 周期的状态。转换参数是 alpha 和 rho。从状态 1 直接进入状态 3 的概率为 0，而状态 3 是吸收状态。

Command:svar(OPTIONS...);

每条马尔科夫链可以控制一组参数的切换。允许参数按方程和方差或斜率和截距划分。

选项

coefficients：指定给定方程中仅斜率和截距受给定链控制。coefficients 或 variances 必须出现一个，但不能同时出现。默认值：none。

variances：指定给定方程中仅方差受给定链控制。coefficients 或 variances 必须出现一个，但不能同时出现。默认值：none。

equations：定义由给定链控制的方程。如果未指定，则所有方程都由 chain 控制。默认值：none。

chain=INTEGER：指定由 *markov_switching* 定义的马尔科夫链。默认值：none。

Command:sbvar(OPTIONS...);

为了有据可查，参见百科：https://archives.dynare.org/DynareWiki/SbvarOptions。

选项

datafile,freq,initial_year,initial_subperiod,final_year,final_subperiod,data,vlist,vlistlog,vlistper,restriction_fname,nlags,cross_restrictions,contemp_reduced_form,real_pseudo_forecast,no_bayesian_prior,dummy_obs,nstates,indxscalesstates,alpha,beta,gsig2_lmdm,q_diag,flat_prior,ncsk,nstd,ninv,indxparr,indxovr,aband,indxap,apband,indximf,indxfore,foreband,indxgforhat,indxgimfhat,indxestima,indxgdls,eq_ms,cms,ncms,eq_cms,tlindx,tlnumber,cnum,forecast,

coefficients_prior_hyperparameters

Block:svar_identification;

模块由 end;终止，包含以下形式的行：

```
UPPER_CHOLESKY;
LOWER_CHOLESKY;
EXCLUSION CONSTANTS;
EXCLUSION LAG INTEGER;EQUATION INTEGER,VARIABLE_NAME[,VARIAB
LE_NAME...];
RESTRICTION EQUATION INTEGER,EXPRESSION=EXPRESSION;
```

为了有据可查，参见百科：https://archives.dynare.org/DynareWiki/SbvarOptions。

Command:ms_estimation(OPTIONS...);

触发马尔科夫转换 SBVAR 模型的初始化文件的创建和估计。运行结束时，A^0、A^+、Q和ζ矩阵包含在 oo_.ms 结构中。

一般选项

file_tag=FILENAME：与此次运行关联的文件名部分。这将创建模型初始化文件 init_<file_tag>.dat。默认值：<mod_file>。

output_file_tag=FILENAME：分配给此次运行的输出文件名部分。这将在其他文件中创建 est_final_<output_file_tag>.out、est_intermediate_<output_file_tag>.out。默认值：<file_tag>。

no_create_init：不要为模型创建初始化文件。传递此选项将导致忽略 *Initializatio n Optitons*。此外，模型将从与先前估计运行相关的输出文件生成（即 est_final_<file_tag>.out、est_intermediate_<file_tag>.out 或 init_<file_tag>.dat，按顺序搜索）。此功能可用于继续先前的估计运行以确保达到收敛，或重新使用初始化文件。注意：如果未传递此选项，则先前估算运行的文件将被覆盖。默认值：off（即创建初始化文件）。

初始化选项

coefficients_prior_hyperparameters=[DOUBLE1 DOUBLE2...DOUBLE6]：设置模型的超参数。参数向量的六个元素有以下解释：

1：A^0和A^+的整体紧性；

2：A^+的相对紧性；

3：常数项的相对紧性；；

4：滞后衰减的紧性（范围：1.2-1.5）；更快的衰减会产生更好的膨胀过程；

5：系数虚拟观测值（单位根）的 nvar 总和的权重；

6：单个虚拟初始观察的权重，包括常数；；

默认值：[1.0 1.0 0.1 1.2 1.0 1.0]

freq=INTEGER|monthly|quarterly|yearly：数据频率（例如 monthly，12）。默认值：4。

initial_year=INTEGER：第一年数据。默认值：none。

initial_subperiod=INTEGER：第一时期数据（即对于季度数据，[1,4]中的整数）。默认值：1。

final_year=INTEGER：最后一年数据。默认值：设置为包含整个数据集。

final_subperiod=INTEGER：末期数据（即对于月度数据，[1,12]中的整数。默认值：当 final_year 也缺失时，设置为包含整个数据集；当指定 final_year 时，设置为给定频率的最大子周期数（即季度数据为 4，月度数据为 12，...）。

datafile=FILENAME：参见 *datafile*。

xls_sheet=QUOTED_STRING：参见 *xls_sheet*。

xls_range=RANGE：参见 *xls_range*。

nlags=INTEGER：模型的滞后数。默认值：1。

cross_restrictions：使用交叉A^0和A^+约束。默认值：off。

contemp_reduced_form：使用同期递归简化形式。默认值：off。

no_bayesian_prior：不使用贝叶斯先验。默认值：off（即使用贝叶斯先验）。

alpha=INTEGER：平方时变结构冲击 lambda 的 Alpha 值。默认值：1。

beta=INTEGER：平方时变结构冲击 lambda 的 Beta 值。默认值：1。

gsig2_lmdm=INTEGER：SimsZha 约束下每个独立λ参数的方差。默认值：50^2。

specification=sims_zha|none：控制了如何施加约束以减少参数数量。默认值：Random Walk。

估计选项

convergence_starting_value=DOUBLE：收敛容差标准，指的是目标函数值的变化。它应该是相当宽松的，因为在估计过程中逐渐收紧。默认值：1e-3。

convergence_ending_value=DOUBLE：收敛标准结束值。远小于平方根机械极小值的值可能矫枉过正。默认值：1e-6。

convergence_increment_value=DOUBLE：确定收敛标准从初值移动到末值的速度。默认值：0.1。

max_iterations_starting_value=INTEGER：爬山优化程序中允许的最大迭代次数，应该相当小，因为会在估计过程中逐渐增加。默认值：50。

max_iterations_increment_value=DOUBLE：确定最大迭代次数增加的速度。默认值：2。

max_block_iterations=INTEGER：参数分块并且在每个模块上优化。执行一组

分块优化后，检查收敛标准，如果违反该标准，则重复分块优化。这控制了可执行分块优化的最大次数。注意，在逐块优化收敛后，会在更新收敛值和最大迭代次数之前对所有参数执行一次优化。默认值：100。

max_repeated_optimization_runs=INTEGER：按照 *max_block_iterations* 描述的整个过程重复，直到优化停止。这是允许过程重复的最大次数。将此设置为 0 将不允许重复。默认值：10。

function_convergence_criterion=DOUBLE：当 max_repeated_optimizations_runs 为正时目标函数的收敛标准。默认值：0.1。

parameter_convergence_criterion=DOUBLE：当 max_repeated_optimizations_runs 为正时参数值的收敛标准。默认值：0.1。

number_of_large_perturbations=INTEGER：按照 *max_block_iterations* 描述的整个过程使用从后验中抽取的随机起始值重复。这指定了使用的随机起始值的数量。设置为 0 则不使用随机起始值。应指定更大数字确保已覆盖整个参数空间。默认值：5。

number_of_small_perturbations=INTEGER：在大扰动停止改善后要执行的小扰动数量。设置的数字远高于 10 可能太大了。默认值：5。

number_of_posterior_draws_after_perturbation=INTEGER：产生小扰动时要执行的连续后验抽取的次数。因为后验抽取是连续相关的，小数字会导致小扰动。默认值：1。

max_number_of_stages=INTEGER：重复小和大扰动，直到改进停止。这指定了允许的最大阶段数。默认值：20。

random_function_convergence_criterion=DOUBLE：number_of_large_perturbations 为正时目标函数的收敛标准。默认值：0.1。

random_parameter_convergence_criterion=DOUBLE：number_of_large_perturbations 为正时参数值的收敛标准。默认值：0.1。

示例

```
ms_estimation(datafile=data,initial_year=1959,final_year=2005,nlags=4,max_repeated_optimization_runs=1,max_number_of_stages=0);

ms_estimation(file_tag=second_run,datafile=data,initial_year=1959,final_year=2005,nlags=4,max_repeated_optimization_runs=1,max_number_of_stages=0);

ms_estimation(file_tag=second_run,output_file_tag=third_run,
```

```
no_create_init,max_repeated_optimization_runs=5,number_of_la
rge_perturbations=10);
```

Command:ms_simulation;

Command:ms_simulation(OPTIONS...);

模拟马尔科夫转换 SBVAR 模型。

选项

file_tag=FILENAME：与 ms_estimation 运行关联的文件名部分。默认值：<mo d_file>。

output_file_tag=FILENAME：分配给此运行的输出文件名部分。默认值：<file _tag>。

mh_replic=INTEGER：要保存的抽取次数。默认值：10,000。

drop=INTEGER：预热处理的抽取次数。默认值：0.1*mh_replica*thinning_f actor。

thinning_factor=INTEGER：抽取总数等于 thinning_factor*mh_replica+ drop。默认值：1。

adaptive_mh_draws=INTEGER：Metropolis-Hastings 抽取的调整期。默认值：30, 000。

save_draws：将A^0、A^+、Q和ζ的所有元素保存到名为 draws_<<file_tag>>.ou t 的文件中，每次在单独的行上抽取。描述这些矩阵如何布局的文件包含于 draws_head er_<<file_tag>>.out。提供了一个名为 load_flat_file.m 的文件，以简化在 MA TLAB/Octave 工作区加载保存的文件到相应变量 A0、Aplus、Q 和 Zeta。默认值：off。

示例

```
ms_simulation(file_tag=second_run);
ms_simulation(file_tag=third_run,mh_replic=5000,thinning_fac
tor=3);
```

Command:ms_compute_mdd;

Command:ms_compute_mdd(OPTIONS...);

根据后验抽样计算马尔科夫转换 SBVAR 模型的边际数据密度。运行结束时，Muller 和 Bridged 日志边际密度包含在 oo_.ms 结构。

选项

file_tag=FILENAME：参见 *file_tag*。

output_file_tag=FILENAME：参见 *output_file_tag*。

simulation_file_tag=FILENAME：与模拟运行关联的文件名部分。默认值：<f ile_tag>。

proposal_type=INTEGER：建议类型：

1：高斯函数；

2：幂函数；

3：截断的幂函数；

4：step 函数；

5：截断的高斯函数。

默认值：3。

proposal_lower_bound=DOUBLE：概率下限。不用于[1,2]中的 proposal_type。所有其他建议类型都是必需的。默认值：0.1。

proposal_upper_bound=DOUBLE：概率的上限。不用于等于 1 的 proposal_type。所有其他建议类型都是必需的。默认值：0.9。

mdd_proposal_draws=INTEGER：建议的抽取次数。默认值：100,000。

mdd_use_mean_center：使用后验均值作为中心。默认值：off。

Command:ms_compute_probabilities;

Command:ms_compute_probabilities(OPTIONS...);

计算马尔科夫转换 SBVAR 模型的平滑状态概率。输出*.eps 文件包含在<output_file_tag/Output/Probabilities>中。

选项

file_tag=FILENAME：参见 *file_tag*。

output_file_tag=FILENAME：参见 *output_file_tag*。

filtered_probabilities：计算滤波的概率而不是平滑。默认值：off。

real_time_smoothed：基于$0 \le t \le nobs$时间 t 信息计算平滑概率。默认值：off。

Command:ms_irf;

Command:ms_irf(OPTIONS...);

计算马尔科夫转换 SBVAR 模型的脉冲响应函数。输出*.eps 文件包含在<output_file_tag/Output/IRF>中，而数据文件包含于<output_file_tag/IRF>中。

选项

file_tag=FILENAME：参见 *file_tag*。

output_file_tag=FILENAME：参见 *output_file_tag*。

simulation_file_tag=FILENAME：参见 *simulation_file_tag*。

horizon=INTEGER：预测范围。默认值：12。

filtered_probabilities：使用样本末尾的滤波概率作为状态概率的初始条件。只能通过 filtered_probabilities、regime 和 regimes 之一。默认值：off。

error_band_percentiles=[DOUBLE1...]：计算的百分位数。默认值：[0.16

0.50 0.84]。如果传递 median，则默认值为[0.5]。

shock_draws=INTEGER：抽取的状态路径数。默认值：10,000。

shocks_per_parameter=INTEGER：在参数不确定性下抽取的状态路径数。默认值：10。

thinning_factor=INTEGER：仅使用后验抽样文件中抽取的 1/thinning_factor。默认值：1。

free_parameters=NUMERICAL_VECTOR：初始化模型 theta 的自由参数向量。默认值：使用估计参数。

parameter_uncertainty：计算参数不确定性下的 IRF。要求已运行 ms_simulation。默认值：off。

regime=INTEGER：给定数据和模型参数，处于指定状态的遍历概率是多少。只能通过 filtered_probabilities、regime 和 regimes 之一。默认值：off。

regimes：描述状态的演变。只能通过 filtered_probabilities、regime 和 regimes 之一。默认值：off。

median：设置 error_band_percentiles=[0.5]的快捷方式。默认值：off。

Command:ms_forecast;

Command:ms_forecast(OPTIONS...);

为马尔科夫转换 SBVAR 模型生成预测。输出*.eps 文件包含于<output_file_tag/Output/Forecast>中，数据文件包含于<output_file_tag/Forecast>。

选项

file_tag=FILENAME：参见 *file_tag*。

output_file_tag=FILENAME：参见 *output_file_tag*。

simulation_file_tag=FILENAME：参见 *simulation_file_tag*。

data_obs_nbr=INTEGER：输出中包含的数据点数。默认值：0。

error_band_percentiles=[DOUBLE1...]：参见 *error_band_percentiles*。

shock_draws=INTEGER：参见 *shock_draws*。

shocks_per_parameter=INTEGER：参见 *shocks_per_parameter*。

thinning_factor=INTEGER：参见 *thinning_factor*。

free_parameters=NUMERICAL_VECTOR：参见 *free_parameters*。

parameter_uncertainty：参见 *parameter_uncertainty*。

regime=INTEGER：参见 *regime*。

regimes：参见 *regimes*。

median：参见 *median*。

horizon=INTEGER：参见 *horizon*。

Command:ms_variance_decomposition;

Command:ms_variance_decomposition(OPTIONS...);

计算马尔科夫转换 SBVAR 模型的方差分解。输出*.eps 文件包含于<output_file_tag/Output/Variance_Decomposition>，而数据文件包含于<output_file_tag/Variance_Decomposition>。

选项

file_tag=FILENAME：参见 *file_tag*。

output_file_tag=FILENAME：参见 *output_file_tag*。

simulation_file_tag=FILENAME：参见 *simulation_file_tag*。

horizon=INTEGER：参见 *horizon*。

filtered_probabilities：参见 *filtered_probabilities*。

no_error_bands：不要输出百分位误差带（即计算平均值）。默认值：off（即输出误差带）

error_band_percentiles=[DOUBLE1...]：参见 *error_band_percentiles*。

shock_draws=INTEGER：参见 *shock_draws*。

shocks_per_parameter=INTEGER：参见 *shocks_per_parameter*。

thinning_factor=INTEGER：参见 *thinning_factor*。

free_parameters=NUMERICAL_VECTOR：参见 *free_parameters*。

parameter_uncertainty：参见 *parameter_uncertainty*。

regime=INTEGER：参见 *regime*。

regimes：参见 *regimes*。

4.24 尾声变量

Block:epilogue;

尾声模块可用于计算模型中不必定义但是感兴趣的输出变量（例如，各种实际/名义份额或相对价格，或季度模型中的年化变量），在计算效率和灵活性方面提供如下优势：

- 运行平滑因子/模拟之后，在尾声模块计算变量，而不需要添加新的定义和方程，也不需要重新运行平滑因子/模拟。甚至后验平滑因子绘制也可以循环用于计算尾声变量，而无需使用新定义和方程重新运行 Sub-draw。
- 减少数据滤波/平滑中的状态空间维度。例如，希望年化变量作为输出。如果在季度模型中定义年增长率，则需要滞后至相关季度变量的七阶；在大/中型模型中，这只会增加状态维度并大幅延长平滑因子的计算时间。

epilogue 模块在 end;终止，并包含以下形式的行：

NAME=EXPRESSION;

示例

```
epilogue;
//annualized level of y
ya=exp(y)+exp(y(-1))+exp(y(-2))+exp(y(-3));
//annualized growth rate of y
gya=ya/ya(-4)-1;
end;
```

4.25 半结构模型+

Dynare 提供了半结构模型的工具，与 FRB/US 模型（参见 Brayton 和 Tinsley，1996）不同，半结构模型的预期未必一致，而是基于 VAR 辅助模型。下文假设每个方程写成 VARIABLE=EXPRESSION 或 T(VARIABLE)=EXPRESSION，其中 T(VARIABLE)表示一个内生变量（log 或 diff）转换。这种表示方法，即每个方程都决定了 LHS 的内生变量，可以在模拟模型时利用（参见 *solve_algo* 中的算法 12 和 14），也是定义计算期望值的辅助模型所必须的（见下文）。

4.25.1 辅助模型

本节定义的两个辅助模型是形成预期的线性后向模型。两个模型都可以被重塑为 VAR(1)过程，因此提供了指定预期过程的同构方式，但在指定协整和误差修正等特征的便利性有所不同。var_model 直接指定了一个 VAR，而 trend_component_model 允许定义一个趋势目标，内生变量在长期内可能被吸引到这个目标上（即一个误差修正模型）。

Command:var_model(OPTIONS...);

model 模块中挑选方程形成一个 VAR 模型。这个模型可以作为 var_expectation_model 或 pac_model 的辅助模型，必须是以下形式：

$$Y_t = \mathbf{c} + \sum_{i=1}^{p} A_i Y_{t-i} + \varepsilon_t$$

$$A_0 Y_t = \mathbf{c} + \sum_{i=1}^{p} A_i Y_{t-i} + \varepsilon_t$$

如果 VAR 是结构性的（见下文），其中Y_t和ε_t是$n \times 1$向量，$\mathbf{c}$是$n \times 1$参数向量，$A_i(i = 0, \dots, p)$是$n \times n$的参数矩阵，A_0是非奇异方阵。向量$\mathbf{c}$和矩阵$A_i(i = 0, \dots, p)$由 Dynare 通过解析 model 模块的方程设定。然后，Dynare 为$\mathcal{Y}_t = \left(1, Y_t, \dots, Y_{t-p+1}\right)'$建立一个 VAR(1)伴随形式的模型，如：

$$\begin{pmatrix} 1 \\ Y_t \\ Y_{t-1} \\ \vdots \\ \vdots \\ Y_{t-p+1} \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0_n' & \dots & \dots & \dots & 0_n' \\ \mathbf{c} & A_1 & A_2 & \dots & \dots & A_p \\ 0_n & I_n & O_n & \dots & \dots & O_n \\ 0_n & O_n & I_n & O_n & \dots & O_n \\ \vdots & O_n & \ddots & \ddots & \ddots & \vdots \\ 0_n & O_n & \dots & O_n & I_n & O_n \end{pmatrix}}_{\mathcal{C}} \begin{pmatrix} 1 \\ Y_{t-1} \\ Y_{t-2} \\ \vdots \\ \vdots \\ Y_{t-p} \end{pmatrix} + \underbrace{\begin{pmatrix} 0 \\ \varepsilon_t \\ 0_n \\ \vdots \\ \vdots \\ 0_n \end{pmatrix}}_{\epsilon_t}$$

假设已处理简化形式的 VAR（否则，右侧将另外先乘以A_0^{-1}，以获得简化形式）。如果 VAR 没有常数，删除系统的第一行和伴随矩阵$\mathcal{C}$的第一列。Dynare 只保存伴随矩阵，因为这是计算期望所需的唯一信息。

MATLAB/Octave variable:oo_.var.MODEL_NAME.CompanionMatrix

var_model 的简化形式伴随矩阵。

选项

model_name=STRING：VAR 模型的名称，在 var_expectation_model 或 pac_model 中作为一个辅助模型引用。需要一个有效的 Matlab 字段名。

eqtags=[QUOTED_STRING[,QUOTED_STRING[,...]]]：建立 VAR 模型 model 模块中的方程列表（使用方程标签 name 引用）。

structural：默认情况下，VAR 模型不是结构性的，即每个方程必须恰好包含一个同时期变量（在 LHS）。如果提供了 structural 选项，则系统中定义的任何变量都可以在t时期的每个方程中出现。Dynar 在内部 e 会将该模型重写为简化形式的 VAR（对隐含矩阵A_0求逆）。

示例

```
var_model(model_name=toto,eqtags=['X','Y','Z']);

model;

[name='X']
x=a*x(-1)+b*x(-2)+c*z(-2)+e_x;

[name='Z']
z=f*z(-1)+e_z;

[name='Y']
y=d*y(-2)+e*z(-1)+e_y;

end;
```

Command:trend_component_model(OPTIONS...);

选取模型模块中的方程形成趋势成分模型。该模型可以作为 var_expectation_model 或 pac_model 的辅助模型，必须是以下形式：

$$\begin{cases}\Delta X_t = A_0(X_{t-1} - C_0 Z_{t-1}) + \sum_{i=1}^{p} A_i \Delta X_{t-i} + \varepsilon_t \\ Z_t = Z_{t-1} + \eta_t\end{cases}$$

其中X_t和Z_t是内生变量的$n \times 1$和$m \times 1$向量。Z_t定义了趋势目标，如果隐含的误差修正矩阵A_0是负定的，则内生变量X_t将吸引到线性组合$C_0 Z_t$。ε_t和η_t是$n \times 1$和$m \times 1$的外生变量向量，$A_i (i = 0, \dots, p)$是$n \times n$的参数矩阵，C_0是一个$n \times m$矩阵。这个模型重新写成水平值的形式，转化为 VAR(1)模型。设$Y_t = (X_t', Z_t')'$，$\zeta_t = (\varepsilon_t', \eta_t')'$。那么可以得到：

$$Y_t = \sum_{i=1}^{p+1} B_i Y_{t-i} + \zeta_t$$

且：

$$B_1 = \begin{pmatrix} I_n + A_0 + A_1 & -\Lambda \\ O_{m,n} & I_m \end{pmatrix}$$

其中$\Lambda = A_0 C_0$。

对于$i = 2, \dots, p$，有：

$$B_i = \begin{pmatrix} A_i - A_{i-1} & O_{n,m} \\ O_{m,n} & O_m \end{pmatrix}$$

且：

$$B_{p+1} = \begin{pmatrix} -A_p & O_{n,m} \\ O_{m,n} & O_m \end{pmatrix}$$

这个层次上的$VAR(p+1)$可以再次重写为$VAR(1)$伴随模型形式。

MATLAB/Octave variable:oo_.trend_component.MODEL_NAME.CompanionMatrix

Trend_component_model 的简化形式伴随矩阵。

选项

model_name=STRING：趋势成分模型的名称，在 var_expectation_model 或 pac_model 中作为一个辅助模型引用。需要一个有效的 Matlab 字段名。

eqtags=[QUOTED_STRING[,QUOTED_STRING[,...]]]：建立趋势分量模型 model 模块中的方程列表（使用方程标签 name 引用）。

targets=[QUOTED_STRING[,QUOTED_STRING[,...]]]：目标列表，对应向量Z_t中的变量，使用 model 模块中关联方程的方程标签 name 引用。target 必须是 eqtags 的子集。

示例

```
trend_component_model(model_name=toto,eqtags=['eq:x1','eq:x2
```

```
','eq:x1bar','eq:x2bar'],targets=['eq:x1bar','eq:x2bar']);

model;

[name='eq:x1']
diff(x1)=a_x1_0*(x1(-1)-x1bar(-1))+a_x1_0_*(x2(-1)-x2bar(-
1))+a_x1_1*diff(x1(-1))+a_x1_2*diff(x1(-2))++a_x1_x2_1*diff
(x2(-1))+a_x1_x2_2*diff(x2(-2))+ex1;

[name='eq:x2']
diff(x2)=a_x2_0*(x2(-1)-x2bar(-1))+a_x2_1*diff(x1(-1))+a_x2_
2*diff(x1(-2))+a_x2_x1_1*diff(x2(-1))+a_x2_x1_2*diff(x2(-2))
+ex2;

[name='eq:x1bar']
x1bar=x1bar(-1)+ex1bar;

[name='eq:x2bar']
x2bar=x2bar(-1)+ex2bar;

end;
```

4.25.2 VAR 预期

假设希望预测一个变量y_t，并且y_t是变量Y_t向量的一个元素，其运动规律由 VAR(1)模型$\mathcal{Y}_t = \mathcal{C}\mathcal{Y}_{t-1} + \epsilon_t$描述。更一般地说，$y_t$可以是$Y_t$中标量变量的线性组合。存在向量 α 使$y_t = \alpha'\mathcal{Y}_t$（如果$y_t$是$Y_t$中的一个变量，即单位矩阵的某列，或定义线性组合权重的任意向量，α 是一个选择向量）。那么，在 RMSE 最小化的意义上，给定$t-\tau$时的信息集（假定它包括到$t-\tau$时的所有可观测变量$\mathcal{Y}_{\underline{t-\tau}}$），对$y_{t+h}$的最佳预测是：

$$y_{t+h|t-\tau} = \mathbb{E}\left[y_{t+h} \mid \mathcal{Y}_{\underline{t-\tau}}\right] = \alpha\mathcal{C}^{h+\tau}\mathcal{Y}_{t-\tau}$$

半结构模型中，出现在$t+h$的变量（例如，动态 IS 曲线中的预期产出缺口或新凯恩斯）菲利普斯曲线中的预期通胀）将被辅助 VAR 模型隐含的预期所取代。另一个情况是计算永久收入。通常情况下，消费将取决于以下方程：

$$\sum_{h=0}^{\infty} \beta^h y_{t+h|t-\tau}$$

假设$0 < \beta < 1$和已知几何级数的极限，可以根据相同的辅助模型求出变量的条件期望：

$$\mathbb{E}\left[\sum_{h=0}^{\infty}\beta^h y_{t+h} \mid \underline{\mathcal{Y}_{t-\tau}}\right] = \alpha\mathcal{C}^{\tau}(I-\beta\mathcal{C})^{-1}\mathcal{Y}_{t-\tau}$$

更一般地，可以考虑有限贴现和。

Command:var_expectation_model(OPTIONS...);

声明用于预测在$t+h$的内生变量或变量线性组合的模型。更一般地，相同模型可用于预测变量的贴现流量或变量的线性表达式：

$$\sum_{h=a}^{b}\beta^{h-\tau}\mathbb{E}[y_{t+h} \mid \underline{\mathcal{Y}_{t-\tau}}]$$

其中$(a,b)\in\mathbb{N}^2$，$a<b$，$\beta\in(0,1]$是一个折现因子，τ是一个有限的正整数。

选项

model_name=STRING：基于期望模型的 VAR 名称，在 model 模块中引用。

auxiliary_model=STRING：关联辅助模型的名称，用 var_model 或 trend_component_model 定义。

expression=STRING|QUOTED_STRING：预期的变量或表达式（变量的线性组合）的名称。如果需要表达式，那么必须使用引号。

discount=PARAMETER_NAME|DOUBLE：折现因子（β）。

horizon=INTEGER|[INTEGER:INTEGER]：水平h的上限b（此时$a=0$），或计算贴现和的周期$a{:}\,b$的范围（上限可以是 Inf）。

time_shift=INTEGER：信息集的位移（τ）。默认值：0。

Operator:var_expectation (NAME_OF_VAR_EXPECTATION_MODEL);

使用此运算符代替超前变量，例如 X(1)，在 model 模块中用基于 VAR 模型的预测替换模型一致预测。

示例

```
var_model(model_name=toto,eqtags=['X','Y','Z']);


var_expectation_model(model_name=varexp,expression=x,auxiliary_model_name=toto,horizon=1,discount=beta);


model;


[name='X']
x=a*x(-1)+b*x(-2)+c*z(-2)+e_x;


[name='Z']
```

```
z=f*z(-1)+e_z;


[name='Y']
y=d*y(-2)+e*z(-1)+e_y;


foo=.5*foo(-1)+var_expectation(varexp);


end;
```

此例的 var_expectation(varexp)代表 x 的超前一步期望值，替代 x(1)。

MATLAB/Octave command:var_expectation.initialize(NAME_OF_VAR_EXPECTATION_MODEL);

建立关联辅助变量模型的伴随矩阵初始化 *var_expectation_model*。需要在尝试模拟或估计模型之前执行。

MATLAB/Octave command:var_expectation.update(NAME_OF_VAR_EXPECTATION_MODEL);

更新/计算 *var_expectation_model* 的简化式参数。需要在尝试模拟或估计模型之前执行，并且要求辅助的 *var_model* 之前已经初始化。

示例(续)

```
var_expectation.initialize('varexp');


var_expectation.update('varexp');
```

警告：对底层辅助 *var_model* 参数的修改需要调用 var_expectation.initialize 和 var_expectation.update 才能生效。对 var_expectation_model 或其相关参数的改变需要调用 var_expectation.update。

4.25.3 PAC 方程

在最简形式，PAC 方程将感兴趣的变量y的变化分解为三个贡献：（1）滞后于目标$y^\star$的偏差；（2）变量y的滞后变化；（3）目标$y^\star$的预期变化。

$$\Delta y_t = a_0(y^\star_{t-1} - y_{t-1}) + \sum_{i=1}^{m-1} a_i \Delta y_{t-i} + \sum_{i=0}^{\infty} d_i \Delta y^\star_{t+i} + \varepsilon_t$$

Brayton 等（2000）展示了如何从二次成本函数的最小化中推导出这样一个方程，该二次成本函数惩罚与目标的预期偏差和y的非平稳性，其中未来成本被贴现（贴现因子为β）。他们还表明，参数$(d_i)_{i\in N}$是m参数a_i和贴现因子β的非线性函数。为了模拟或估计这个方程，我们需要弄清楚如何确定目标的预期变化。这可以像上一节那样使用基于 VAR 的预期，或者考虑模型一致的预期（MCE）。

为了确保内生变量y在（确定性的）长期等于目标$y^\star$，即误差修正项在长期为零，可以选择在方程中加入一个增长中性修正。假设g是长期增长率，对于y和$y^\star$，长期（假设数据取对数）必有：

$$g = a_0(y_\infty^\star - y_\infty) + g\sum_{i=1}^{m-1} a_i + g\sum_{i=0}^{\infty} d_i$$
$$\Leftrightarrow a_0(y_\infty^\star - y_\infty) = \left(1 - \sum_{i=1}^{m-1} a_i - \sum_{i=0}^{\infty} d_i\right) g$$

除非对系数$(a_i)_{i=0}^{m-1}$施加额外的约束，即最小化成本函数，否则没有理由使右边为零。相反，可以选择性地将右边添加到 PAC 方程中，确保误差修正项渐进为零。

PAC 方程可以通过增加外生变量生成。通过两种不排斥的方式实现：用原始 PAC 方程（来自于一个优化程序）和一个包含外生变量的线性表达式（被称为经验法则部分，而不是从成本函数最小化得出的部分；不能与外生冲击混淆）的凸组合来代替 PAC 方程：

$$\Delta y_t = \lambda\left(a_0(y_{t-1}^\star - y_{t-1}) + \sum_{i=1}^{m-1} a_i \Delta y_{t-i} + \sum_{i=0}^{\infty} d_i \Delta y_{t+i}^\star\right) + (1-\lambda)\gamma' X_t + \varepsilon_t$$

其中$\lambda \in [0,1]$是纯 PAC 方程的权重，γ是参数的$k \times 1$向量，X_t是经验法则部分$k \times 1$变量向量。或者可以简单地将外生变量添加到 PAC 方程中（没有权重λ）：

$$\Delta y_t = a_0(y_{t-1}^\star - y_{t-1}) + \sum_{i=1}^{m-1} a_i \Delta y_{t-i} + \sum_{i=0}^{\infty} d_i \Delta y_{t+i}^\star + \gamma' X_t + \varepsilon_t$$

Command:pac_model(OPTIONS...);

声明 PAC 模型。一个*.mod 文件可以有一个以上的 PAC 模型或 PAC 方程，每个 PAC 方程必须关联一个不同的 PAC 模型。

选项

model_name=STRING：PAC 模型的名称，在 model 模块引用。

auxiliary_model=STRING：用 var_model 或 trend_component_model 定义关联辅助模型的名称，计算基于 VAR 的目标预期变化的期望，即评估$\sum_{i=0}^{\infty} d_i \Delta y_{t+i}^\star$。然后，无限和将被辅助模型的伴随表示中涉及变量的线性组合取代。在 PAC 方程定义线性组合的权重是$(a_i)_{i=0}^{m-1}$系数的非线性函数。如果 Dynare 不理解目标预期变化必须在 MCE 假设下进行计算，那么这个选项就不是强制性的。这是通过递归重写 Brayton 等（2000）的方程 10 中所示的无限和来实现的。

discount=PARAMETER_NAME|DOUBLE：出现在成本函数定义中的未来预期成本的贴现因子（β）。

growth=PARAMETER_NAME|VARIABLE_NAME|EXPRESSION|DOUBLE：如果存在，则在 PAC 方程中加入增长中性修正。用户必须确保提供的值（或长期水平，如果给出了一个变量或表达式）与内生变量的渐进增长率一致。

Operator:pac_expectation (NAME_OF_PAC_MODEL);

在 model 模块定义的 PAC 方程中，使用此运算符代替无限和$\sum_{i=0}^{\infty} d_i \Delta y_{t+i}^{\star}$。根据期望形成假设，它将被辅助模型的伴随表示中涉及变量的线性组合或递归前向方程取代。

MATLAB/Octave command:pac.initialize(NAME_OF_PAC_MODEL);

MATLAB/Octave command:pac.update(NAME_OF_PAC_MODEL);

与上一节中的 VAR 期望相同，通过构建辅助模型的伴随矩阵，初始化 PAC 模型，并计算 PAC 方程的简化形式参数（辅助模型的伴随表示中涉及变量的线性组合中的权重，或 MCE 情况下无限和的递归表示中的参数）。

示例

```
trend_component_model(model_name=toto,eqtags=['eq:x1','eq:x2
','eq:x1bar','eq:x2bar'],targets=['eq:x1bar','eq:x2bar']);

pac_model(auxiliary_model_name=toto,discount=beta,growth=dif
f(x1(-1)),model_name=pacman);

model;

[name='eq:y']
y=rho_1*y(-1)+rho_2*y(-2)+ey;

[name='eq:x1']
diff(x1)=a_x1_0*(x1(-1)-x1bar(-1))+a_x1_1*diff(x1(-1))+a_x1_
2*diff(x1(-2))+a_x1_x2_1*diff(x2(-1))+a_x1_x2_2*diff(x2(-2))
+ex1;

[name='eq:x2']
diff(x2)=a_x2_0*(x2(-1)-x2bar(-1))+a_x2_1*diff(x1(-1))+a_x2_
2*diff(x1(-2))+a_x2_x1_1*diff(x2(-1))+a_x2_x1_2*diff(x2(-2))
+ex2;

[name='eq:x1bar']
x1bar=x1bar(-1)+ex1bar;

[name='eq:x2bar']
```

```
x2bar=x2bar(-1)+ex2bar;


[name='zpac']
diff(z)=e_c_m*(x1(-1)-z(-1))+c_z_1*diff(z(-1))+c_z_2*diff(z
(-2))+pac_expectation(pacman)+ez;


end;


pac.initialize('pacman');


pac.update.expectation('pacman');
```

4.25.4 PAC 方程估计

上一节介绍的 PAC 方程可以估计。这个方程对于估计的参数$(a_i)_{i=0}^{m-1}$来说是非线性的，因为简化形式的参数（在计算无限和的时候）是自回归参数和误差修正参数的非线性函数。Brayton 等（2000）展示了如何通过迭代 OLS 估计 PAC 方程。虽然这种方法是在 Dynare 中实现的，但主要是为了比较，我们也提出了 NLS 估计，这是更可取的（NLS 的渐进特性有更坚实的基础）。

注意，目前使用如贝叶斯技术等联合估计 PAC 方程和模型的其余参数是不可行的。因此，PAC 方程的估计只能以辅助模型的参数值为条件进行。

警告：下面描述的估计程序需要将选项 `json=compute` 传递给预处理器（通过命令行或在`*.mod` 文件的顶部，参见 3.1 Dynare 调用）。

MATLAB/Octave command:pac.estimate.nls(EQNAME,GUESS,DATA,RANGE[,ALGO]);

MATLAB/Octave command:pac.estimate.iterative_ols(EQNAME,GUESS,DATA,RANGE);

触发 PAC 方程的 NLS 或迭代 OLS 估计。`EQNAME` 是一个指定待估 PAC 方程的行字符数组（PAC 方程必须有一个用方程标签指定的名称）。`DATA` 是一个 `dseries` 对象，包含估计所需的数据（即方程中所有内生和外生变量的数据）。PAC 方程的残差值（对应定义的 `varexo`）也必须是 `DATA` 的成员，但要用 `NaN` 值填充。`RANGE` 是一个 `dates` 对象定义样本的时间跨度。`ALGO` 是一个行字符数组，为 `NLS` 选择方法（或最小化算法）。可能的值是：`fmincon`、`fminunc`、`fminsearch`、`lsqnonlin`、`pileswarm`、`csminwel`、`simplex`、`annealing` 和 `GaussNewton`。前四种算法需要 Mathworks 的优化工具箱。第五种算法需要 Mathworks 全局优化工具箱。当优化算法允许时，我们对误差修正参数施加约束，该参数必须是小于 `1` 的正数（`fmincon`、`lsqnonlin`、`pileswarm` 和 `anneal`

ing 就是这种情况)。默认优化算法是 csminwel。GUESS 是一个包含估计参数初始猜测值的结构。每个字段都是 PAC 方程中一个参数的名称，并保存参数的初始猜测值。如果有些参数校准过，那么它们就不应该是 GUESS 结构的成员（调用估计程序之前必须在*.mod 文件中提供数值)。

对于 NLS 例程，估计后的估计结果以表格形式显示。对于 NLS 和迭代 OLS 例程，结果都保存在 oo_中（在字段 NLS 或 iterative_ols 下)。此外，参数值在 M_.params 更新。

示例（续）

```
//Set the initial guess for the estimated parameters
eparams.e_c_m=.9;
eparams.c_z_1=.5;
eparams.c_z_2=.2;

//Define the dataset used for estimation
edata=TrueData;
edata.ez=dseries(NaN);//Set to NaN the residual of the equat
ion.

pac.estimate.nls('zpac',eparams,edata,2005Q1:2005Q1+200,'ann
ealing');
```

警告：*GUESS* 和 *DATA* 的规范涉及结构的使用。因此，它们的子字段不会在 Dynare 运行中清除，因为结构会留在工作区中。例如更改校准参数时，小心从内存中清除这些结构（例如在*.mod 文件中)。

4.26 显示和保存结果

Dynare 对于绘制模拟结果和保存结果提出了命令。

Command:rplot VARIABLE_NAME...;

绘制一个或多个变量的模拟路径。以 perfect_foresight_solver，simul（参见 4.12 确定性模拟）或带有 periods 的 stoch_simul（参见 4.13 随机解和模拟）形式存储在 oo_.endo_simul 中。变量以水平值表示。

Command:dynatype(FILENAME)[VARIABLE_NAME...];

在名为 FILENAME 的文本文件中打印列出的内生变量或外生变量。如果没有列出 VARIABLE_NAME，所有内生变量都会打印出来。

Command:dynasave(FILENAME)[VARIABLE_NAME...];

列出的内生或外生变量保存在名为 FILENAME 的二进制文件中。如果没有列出 VARI

ABLE_NAME，所有内生变量都会被保存。

在 MATLAB 或 Octave 中，可以用以下命令检索使用 dynasave 命令保存的变量：

```
load(FILENAME,'-mat')
```

4.27 宏处理语言

在*.mod 文件中可以使用“macro”命令来执行以下任务：模块化源文件、循环复制方程块、有条件地执行一些代码、在方程式中写入指标和或积等。Dynare 宏语言提供了一组新的可以应用到*.mod 文件的宏命令，特征如下：

- 文件收录；
- 循环（for 结构）；
- 条件包含（if/then/else 结构）；
- 表达式替换

从技术上讲，宏语言完全独立于基础 Dynare 语言，并由 Dynare 预处理器的一个单独组件处理。宏处理器转换*.mod 文件，将宏放入不带宏的*.mod 文件中（执行扩展/包含），然后提供给 Dynare 解析器。理解的关键是宏处理器只执行文本替换（类似于 C 预处理程序或 PHP 语言）。注意，可以使用 dynare 命令的 savemacro 选项查看宏处理器的输出（参见 3.1 Dynare 调用）。

宏处理器是通过在*.mod 文件中放置宏指令被调用的。指令以@符号为前缀，后面紧跟@#。它们不产生输出，而是向宏处理器发出指令。在大多数情况下，指令只占用一行文本。在必要的情况下，行尾的两个反斜杠（\\）表示下一行继续执行指令。宏处理器不会解释//后面的宏指令。由于历史原因，注释模块中的指令（即被/*和*/包围）由宏处理器解释。用户不应该依赖于此行为。主要指令如下：

- @#includepath：用来搜索要包含的文件的路径；
- @#include：用于文件收录；
- @#define：用于定义宏处理器变量；
- @#if、@#ifdef、@#ifndef、@#elseif、@#else、@#endif：用于有条件的陈述；
- @#for、@#endfor：用于构造循环。

宏处理器维护自己的变量列表（不同于模型变量和 MATLAB/Octave 变量）。这些宏变量使用@#defined 指令分配，并且可以是以下类型：布尔值、实数、字符串、元组、函数和数组（任何先前类型）。

4.27.1 宏表达式

宏表达式可以用在两个地方：

- 直接在宏指令中；
- 在*.mod 文件的正文中，在符号和花括号之间（如@{expr}）：宏处理器将用其值

替换表达式。

构造赋值给宏变量或在宏指令中使用的宏表达式是可能的。表达式使用基本类型（布尔值、实数、字符串、元组、数组）、内涵式、宏变量、宏函数和标准运算符等符号构造。

注意：手册其他位置的 MACRO_EXPRESSION 表示按照本节中的解释构造表达式。

布尔值

以下运算符可用于布尔值：

- 比较运算符：==、!=
- 逻辑运算符：&&、||、!

实数

以下运算符可用于实数：

- 算术运算符：+、-、*、/、^
- 比较运算符：<、>、<=、>=、==、!=
- 逻辑运算符：&&、||、!
- 增量为 1 的范围：REAL1:REAL2（例如，1:4 相当于实数数组[1,2,3,4]）。4.6 版更改：以前，将括号括在冒号运算符的参数周围（例如[1:4]）无效用。现在，[1:4]将创建一个包含数组的数组（即[[1,2,3,4]]）。
- 用户定义增量的范围：REAL1:REAL2:REAL3（例如，6:-2.1:-1 等价于实数数组[6,3.9,1.8,-0.3]）。
- 函数：max、min、mod、exp、log、log10、sin、cos、tan、asin、acos、atan、sqrt、cbrt、sign、floor、ceil、trunc、erf、erfc、gamma、lgamma、round、normpdf、normcdf。注意：ln 可代替 log。

字符串

字符串文字必须用双引号括起来（如“name”）。以下运算符可用于字符串：

- 比较运算符：<、>、<=、>=、==、!=
- 两个字符串的串联：+
- 子串的提取：如果 s 是字符串，则 s[3]是只包含 s 的第三个字符的字符串，s[4:6]包含第 4 到 6 个字符
- 函数：length

元组

元组用括号括起来，元素用逗号分隔（如(a,b,c)或(1,2,3)）。以下运算符可用于元组：

- 比较运算符：==、!=
- 函数：empty、length

数组

数组用方括号括起来，元素用逗号分隔（如[1,[2,3],4]或["US","FR"]）。以下运算符可用于数组：

- 比较运算符：==、!=
- 解引用：如果 v 是一个数组，那么 v[2]是第二个元素
- 两个数组的串联：+
- 设置两个数组的并集：|
- 设置两个数组的交集：&
- 差集-：返回第一个操作数，其中第二个操作数的元素已被删除。
- 两个数组的笛卡尔积：*
- 一个数组 N 次的笛卡尔积：^N
- 子数组的提取：例如 v[4:6]
- 测试数组的成员资格：in 运算符（例如：["a","b","c"]中的"b"返回 1）
- 函数：empty、sum、length

内涵式

内涵式语法是一种从其他数组生成数组的速记方式，可以采用三种不同的方式使用内涵式语法：滤波、映射和滤波与映射。

滤波：允许从满足特定条件的数组中选择元素。

示例

创建一个新数组，从数组 1:5 中选择偶数：

```
[i in 1:5 when mod(i,2)==0]
```

将会得到：

```
[2,4]
```

映射：转换数组的每个元素。

示例

创建一个新数组，将数组的所有从 1:5 的元素平方：

```
[i^2 for i in 1:5]
```

将会得到：

```
[1,4,9,16,25]
```

滤波与映射：结合前文算法转换数组的每个选定元素。

示例

创建一个新数组，将数组的所有从 1:5 的偶数元素平方：

```
[i^2 for i in 1:5 when mod(i,2)==0]
```

将会得到：

```
[4,16]
```

进一步的示例

```
[(j,i+1) for (i,j) in (1:2)^2]
[(j,i+1) for (i,j) in (1:2)*(1:2) when i<j]
```

将会得到：

```
[(1,2),(2,2),(1,3),(2,3)]
[(2,2)]
```

函数

可以使用@#define 指令在宏处理器中定义函数（见下文）。函数在调用时被评估，而不是在定义时间。函数可以包含在表达式中，可以与它们组合的运算符取决于返回类型。

检查变量类型

给定一个变量名称或文字，可用以下函数检查计算结果的类型：isboolean、isreal、isstring、istuple 和 isarray。

示例

代码	输出
isboolean(0)	false
isboolean(true)	true
isreal("str")	false

类型之间转换

一种类型的变量和文字可以转换为另一种类型。某些类型更改很简单（例如将实数更改为字符串），而其他类型更改有某些要求（例如，数组转换为实数，必须是包含可以转换为实数的类型的单元素数组）。

示例

代码	输出
(bool)-1.1	true
(bool)0	false
(real)"2.2"	2.2
(tuple)[3.3]	(3.3)
(array)4.4	[4.4]
(real)[5.5]	5.5
(real)[6.6,7.7]	error
(real)"8.8 in a string"	error

可以在表达式中使用强制转换：

示例

代码	输出

(bool)0&&true	false
(real)"1"+2	3
(string)(3+4)	"7"
(array)5+(array)6	[5,6]

4.27.2 宏指令

Macro directive:@#includepath "PATH"

Macro directive:@#includepath MACRO_EXPRESSION

当查找@#include 指定的*.mod 文件时，该指令将 PATH 中包含的路径添加到要搜索的路径列表中。如果提供了 MACRO_EXPRESSION 参数，参数必须计算为字符串。注意，这些路径是在使用-I 传递的任何路径之后添加的。

示例

```
@#includepath "/path/to/folder/containing/modfiles"
@#includepath folders_containing_mod_files
```

Macro directive:@#include "FILENAME"

Macro directive:@#include MACRO_EXPRESSION

该指令只是将另一个文件的内容包含在它的位置上，完全等价于所包含文件内容的复制/粘贴。如果提供了 MACRO_EXPRESSION 参数，则该参数必须计算为字符串。注意，嵌套包含（即从包含的文件中包含一个文件）是有可能的。该文件将在当前目录中搜索。如果找不到，在-I 和@#includepath 提供的文件夹中搜索该文件。

示例

```
@#include "modelcomponent.mod"
@#include location_of_modfile
```

Macro directive:@#define MACRO_VARIABLE

Macro directive:@#define MACRO_VARIABLE=MACRO_EXPRESSION

Macro directive:@#define MACRO_FUNCTION=MACRO_EXPRESSION

定义宏变量或宏函数。

示例 1

```
@#define var                 //Equals 1
@#define x =5                //Real
@#define y ="US"             //String
@#define v =[1,2,4]          //Real array
@#define w =["US","EA"]      //String array
@#define u =[1,["EA"]]       //Mixed array
@#define z =3+v[2]           //Equals 5
```

```
@#define t =("US" in w)     //Equals 1 (true)
@#define f(x)=" "+x+y       //Function `f` with argument `x`
                           //returns the string ' '+x+'US'
```

示例 2

```
@#define x=1
@#define y=["B","C"]
@#define i=2
@#define f(x)=x+"+"+y[i]
@#define i=1

model;
  A=@{y[i]+f("D")};
end;
```

严格意义上等价于：

```
model;
  A=BD+B;
end;
```

Macro directive:@#if MACRO_EXPRESSION

Macro directive:@#ifdef MACRO_VARIABLE

Macro directive:@#ifndef MACRO_VARIABLE

Macro directive:@#elseif MACRO_EXPRESSION

Macro directive:@#else()

Macro directive:@#endif()

有条件地包含*.mod 文件的某些部分。@#if、@#ifdef 或@#ifndef 与下一个@#elseif、@#else 或@#endif 之间的行仅在条件计算结果为 true 时才执行。在@#if 主体之后，可有零个或多个@#elseif 分支。仅当前面的@#if 或@#elseif 条件计算结果为 false 时才计算@#elseif 条件。@#else 分支是可选的，并且仅在所有@#if 和@#elseif 语句的计算结果为 false 时才执行。

注意，使用@#ifdef 时，如果之前已定义 MACRO_VARIABLE，则无论数值如何，条件都将评估为真。相反，如果 MACRO_VARIABLE 尚未定义，@#ifndef 将评估为真；您可以使用已定义的运算符检查是否已定义变量。因此，如果变量 X 尚未定义，进入@#elseif 分支的主体可以编写@#elseif!defined(X)。

注意，如果 MACRO_EXPRESSION 结果出现的实数，被解释为布尔值；0 解释为 false，否则解释为 true。此外，由于实数不精确，在 MACRO_EXPRESSION 测试时必须

格外小心。例如，exp(log(5))==5 将评估为 false。因此，比较实数时，通常在待估数值周围使用零容差，例如 exp(log(5))>5-1e-14&&exp(log(5))<5+1e-14。

示例 1

使用宏观变量在两种替代货币政策规则之间选择：

```
@#define linear_mon_pol=false//0 would be treated the same
...
model;
@#if linear_mon_pol
  i=w*i(-1)+(1-w)*i_ss+w2*(pie-piestar);
@#else
  i=i(-1)^w*i_ss^(1-w)*(pie/piestar)^w2;
@#endif
...
end;
```

这将得到：

```
...
model;
  i=i(-1)^w*i_ss^(1-w)*(pie/piestar)^w2;
...
end;
```

示例 2

使用宏观变量在两种替代货币政策规则之间选择。此示例与前一个示例之间的唯一区别是使用@#ifdef 替代@#if。尽管 linear_mon_pol 包含值 false，因为@#ifdef 仅检查变量是否已定义，但输出线性货币政策：

```
@#define linear_mon_pol=false//0 would be treated the same
...
model;
@#ifdef linear_mon_pol
  i=w*i(-1)+(1-w)*i_ss+w2*(pie-piestar);
@#else
  i=i(-1)^w*i_ss^(1-w)*(pie/piestar)^w2;
@#endif
...
end;
```

这将得到：

```
...
model;
  i=w*i(-1)+(1-w)*i_ss+w2*(pie-piestar);
...
end;
```

Macro directive:@#for MACRO_VARIABLE in MACRO_EXPRESSION

Macro directive:@#for MACRO_VARIABLE in MACRO_EXPRESSION when MACRO_EXPRESSION

Macro directive:@#for MACRO_TUPLE in MACRO_EXPRESSION

Macro directive:@#for MACRO_TUPLE in MACRO_EXPRESSION when MACRO_EXPRESSION

Macro directive:@#endfor()

用于复制*.mod 文件部分的循环构造。注意，此构造可以包含变量/参数声明、计算任务，但不能包含模型声明。

示例 1

```
model;
@#for country in ["home","foreign"]
  GDP_@{country}=A*K_@{country}^a*L_@{country}^(1-a);
@#endfor
end;
```

后者等价于：

```
model;
  GDP_home=A*K_home^a*L_home^(1-a);
  GDP_foreign=A*K_foreign^a*L_foreign^(1-a);
end;
```

示例 2

```
model;
@#for (i,j) in ["GDP"]*["home","foreign"]
  @{i}_@{j}=A*K_@{j}^a*L_@{j}^(1-a);
@#endfor
end;
```

后者等价于：

```
model;
```

```
  GDP_home=A*K_home^a*L_home^(1-a);
  GDP_foreign=A*K_foreign^a*L_foreign^(1-a);
end;
```

示例 3

```
@#define countries=["US","FR","JA"]
@#define nth_co="US"
model;
@#for co in countries when co != nth_co
  (1+i_@{co})=(1+i_@{nth_co})*E_@{co}(+1)/E_@{co};
@#endfor
  E_@{nth_co}=1;
end;
```

后者等价于：

```
model;
  (1+i_FR)=(1+i_US)*E_FR(+1)/E_FR;
  (1+i_JA)=(1+i_US)*E_JA(+1)/E_JA;
  E_US=1;
end;
```

Macro directive:@#echo MACRO_EXPRESSION

要求预处理器在标准输出上显示一些消息。参数必须计算为字符串。

Macro directive:@#error MACRO_EXPRESSION

要求预处理器在标准输出上显示一些错误消息并中止。参数必须计算为字符串。

Macro directive:@#echomacrovars()

Macro directive:@#echomacrovars MACRO_VARIABLE_LIST

Macro directive:@#echomacrovars(save) MACRO_VARIABLE_LIST

要求预处理器显示到目前为止所有宏变量的值。如果传递 save 选项，则宏变量的值将保存到 options_.macrovars_line_<<line_numbers>>。如果传递 NAME_LIST，则只显示/保存具有该名称的变量和函数。

示例

```
@#define A=1
@#define B=2
@#define C(x)=x*2
@#echomacrovars A C D
```

上面命令的输出是：

```
Macro Variables:
  A=1
Macro Functions:
  C(x)=(x*2)
```

4.27.3 典型用法

4.27.3.1 模块化

@#include 指令将*.mod 文件拆分为多个模块化组件。

示例设置：

modeldesc.mod：包含变量的声明，模型方程和冲击声明。

simul.mod：包括 modeldesc.mod，可以校准参数和运行随机模拟。

estim.mod：包括 modeldesc.mod，声明参数的先验值，运行贝叶斯估计。

Dynare 可以调用 simul.mod 和 estim.mod，但是在 modeldesc.mod 上运行没有意义。主要优点是不需要手动复制/粘贴整个模型（在开始时）或更改模型（在开发期间）。

4.27.3.2 乘积的索引化和

下面的示例演示如何构造移动平均值：

```
@#define window=2


var x MA_x;
...
model;
...
MA_x=@{1/(2*window+1)}*(
@#for i in -window:window
        +x(@{i})
@#endfor
       );
...
end;
```

经过宏处理，等价于：

```
var x MA_x;
...
model;
...
MA_x=0.2*(
```

```
      +x(-2)
      +x(-1)
      +x(0)
      +x(1)
      +x(2)
     );
...
end;
```

4.27.3.3 多国模型

这是一个多国模型的基本示例：

```
@#define countries=["US","EA","AS","JP","RC"]
@#define nth_co="US"

@#for co in countries
var Y_@{co} K_@{co} L_@{co} i_@{co} E_@{co} ...;
parameters a_@{co} ...;
varexo ...;
@#endfor

model;
@#for co in countries
 Y_@{co}=K_@{co}^a_@{co}*L_@{co}^(1-a_@{co});
...
@#if co != nth_co
 (1+i_@{co})=(1+i_@{nth_co})*E_@{co}(+1)/E_@{co};//UIP relat
ion
@#else
 E_@{co}=1;
@#endif
@#endfor
end;
```

4.27.3.4 内生化参数

当校准模型稳态时，考虑把参数当作内生变量是有用的（反之亦然）。例如定义 CES 生产函数：

$$y = \left(\alpha^{1/\xi}\ell^{1-1/\xi} + (1-\alpha)^{1/\xi}k^{1-1/\xi}\right)^{\xi/(\xi-1)}$$

GDP 中的劳动份额定义为：

$$lab_rat = (w\ell)/(py)$$

模型中的α是一个（份额）参数，而 lab_rat 是一个内生变量。

显然，校准α不简单，相反，我们有真实世界的 lab_rat 数据，并且很明显这两个变量存在经济联系。

解决方案是使用一种称为变量翻转法，包括改变计算稳态的方式。此时的α是一个内生变量，lab_rat 成为参数。一个经济相关的值校准为 lab_rat，并且求解算法将推导出α的隐含值。

以下文件组成了实施方案：

modeqs.mod：包含变量声明和模型方程。α和 lab_rat 声明的代码格式如下：

```
@#if steady
  var alpha;
  parameter lab_rat;
@#else
  parameter alpha;
  var lab_rat;
@#endif
```

steady.mod：计算稳态，开始于：

```
@#define steady=1
@#include "modeqs.mod"
```

然后初始化参数（包括 lab_rat，不包括α），计算稳态（使用内生性猜测值，包括α），最后在文件中使用 save_params_and_steady_state 命令保存参数值和稳态内生变量值。

simul.mod：计算模拟值，开始于：

```
@#define steady=0
@#include "modeqs.mod"
```

然后利用 load_params_and_steady_state 命令，从文件中加载参数值和稳态内生变量值，并进行仿真计算

4.27.4 循环与宏处理器循环

假设一个参数为ρ的模型，模拟三个数值：$\rho = 0.8,0.9,1$。以下方法可实现：

使用 MATLAB/Octave 循环

```
rhos=[0.8,0.9,1];
for i=1:length(rhos)
```

```
  rho=rhos(i);
  stoch_simul(order=1);
end
```

这里的循环没有展开，MATLAB/Octave 管理迭代次数。有很多迭代时会很有趣。

使用宏处理器循环（案例 1）

```
rhos=[0.8,0.9,1];
@#for i in 1:3
  rho=rhos(@{i});
  stoch_simul(order=1);
@#endfor
```

除了循环是展开的，与前面的例子非常相似。宏处理器管理循环索引，但不管理数据数组（rhos）。

使用宏处理器循环（案例 2）

```
@#for rho_val in [0.8,0.9,1]
  rho=@{rho_val};
  stoch_simul(order=1);
@#endfor
```

这种方法的优点是语法更短，因为值列表直接在循环构造中给出。不便之处在于，不能重复使用存储在 Matlab/Octave 变量中的数组。

4.28 逐句包含

包含在 verbatim 模块的所有内容传递给<mod_file>.m 文件。

Block:verbatim

默认情况下，只要 Dynare 遇到解析器无法理解的代码，它就会直接传递到预处理器输出。为了强制这种行为，可以使用 verbatim 模块。传递给<mod_file>.m 文件的代码包含 Dynare 预处理器识别的标记时，将会很有用。

示例

```
verbatim;
%Anything contained in this block will be passed
%directly to the <modfile>.m file,including comments
var=1;
end;
```

4.29 Misc 命令

Command:set_dynare_seed(INTEGER)

Command:set_dynare_seed('default')

Command:set_dynare_seed('clock')

Command:set_dynare_seed('reset')

Command:set_dynare_seed('ALGORITHM',INTEGER)

设置生成随机数的种子。可以设置给定的整数值、使用默认值或使用时钟（使用后者在不同的 Dynare 运行中将得到不同的结果）。reset 选项用于重置种子为最后一个 set_dynare_seed 命令设置的值。在 MATLAB7.8 或更高版本上，还可以选择生成随机数的特定算法；可接受的值是 mcg16807、mlfg6331_64、mrg32k3a、mt19937ar（默认值）、shr3cong 和 swb2712。

Command:save_params_and_steady_state(FILENAME);

对于所有参数、内生变量和外生变量，使用简单的名称/值关联表将它们的值存储在文本文件中。

- 参数值取自最后一个参数初始值；
- 外生变量值取自最后一个 initval 模块；
- 内生变量值取自上一个稳态计算值（或者，如果没有计算稳态，则取自最后一个 initval 模块）。

注意，文件中没有存储变量类型，因此可以在不同变量类型的设置中使用 load_params_and_steady_state 重载这些值。

这个函数的典型用途是通过校准一些内生变量的稳态值（这意味着一些参数在稳态计算过程中必须内生）来计算模型的稳态。

然后，编写第一个*.mod 文件，使用 save_params_and_steady_state 计算稳态，并在文件末尾保存计算结果。

在第二个设计执行实际模拟的文件中，在变量声明之后使用 load_params_and_steady_state，以便加载先前计算的稳态（包括在稳态计算期间的内生化处理的参数）。

之所以需要两个独立的*.mod 文件，是因为稳态校准和模拟仿真的文件之间的变量声明不同（内生变量和参数的集合在两者之间不同），导致不同的 var 和 parameters 语句，还可以用@#include 指令在两个文件之间共享模型方程（参见 4.27 宏处理语言）。

Command:load_params_and_steady_state(FILENAME);

对于所有的参数、内生变量和外生变量，用 save_params_和_steady_state 创建的文件加载数值。

- 参数值初始化，就像在*.mod 文件中校准了一样；
- 内生变量和外生变量的值初始化，因为可能来自一个 initval 模块。

此函数与 save_params_and_steady_state 一起使用，更多的信息参见该函数的文档说明。

Command:compilation_setup(OPTIONS);

存在 *use_dll* 选项时，Dynare 使用分配的 GCC 编译器编译由预处理器生成的静态和动态 C 文件。您可以使用此选项更改所使用的编译器、标志和库。

选项

compiler=FILENAME：编译器的路径。

substitute_flags=QUOTED_STRING：要使用的标志而不是默认标志。

add_flags=QUOTED_STRING：除了默认标志之外要使用的标志。如果传递 substitute_flags，这些标志添加到指定的标志。

substitute_libs=QUOTED_STRING：要链接的库而不是默认库。

add_libs=QUOTED_STRING：除了默认库之外要链接的库。如果传递 substitute_libs，这些库添加到指定的库。

MATLAB/Octave command:dynare_version;

输出当前正在使用的 Dynare 版本（即在 Matlab/Octave 路径上最高级的版本）。

MATLAB/Octave command:write_latex_definitions;

将模型变量的名称、LaTeX 名称和长名称写入名为<<M_.fname>>_latex_definitions.tex 文件中的表。需要以下 LaTeX 包：longtable。

MATLAB/Octave command:write_latex_parameter_table;

将模型参数的 LaTeX 名称、参数名称和长名称写入名为<<M_.fname>>_latex_parameters.tex 文件中的表。该命令写入当前存储的参数值。因此，如果计算稳态设置或更改了参数，则应在稳态命令之后调用，确保正确更新参数。长名称可以用来添加参数描述。需要以下 LaTeX 包：longtable、booktabs。

MATLAB/Octave command:write_latex_prior_table;

将先验分布的描述性统计信息写入名为<<M_.fname>>_latex_priors_table.tex 文件中的 LaTeX 表。该命令写入当前存储的先验定义中。因此，必须在估计的 estimated_params 模块之后调用此命令。如果先验定义在测量误差之上，也必须先于声明观察变量（使用 varobs）。如果没有定义先验密度（ML 估计）或缺少观测变量的声明将显示警告。需要以下 LaTeX 包：longtable、booktabs。

MATLAB/Octave command:collect_latex_files;

编写一个收集 Dynare 生成的输出，名为<<M_.fname>>_TeX_binder.tex 的 LaTeX 文件到某个文件中。该文件可用 pdflatex 编译，并自动尝试加载所有必需的包。需要下列 LaTeX 包：breqn、psfrag、graphicx、epstopdf、longtable、booktabs、caption、float、amsmath、amsfonts 和 morefloat。

5 配置文件

配置文件用于向 Dynare 提供与模型无关的信息（因此不放在模型文件中）。目前仅在使用 Dynare 运行并行计算时使用。

在 Linux 和 macOS 上，配置文件的默认位置是`$HOME/.dynare`。而在 Windows 上，配置文件是`%APPDATA%\dynare.ini`（典型如 `c:\Users\USERNAME\AppData\dynare.ini`）。可以使用 `dynare` 命令的 `conffile` 选项指定非标准位置（参见 3.1 Dynare 调用）。

配置文件的解析区分大小写，应采取以下形式，每个选项/选择组都放在换行符上：

```
[command0]
option0=choice0
option1=choice1

[command1]
option0=choice0
option1=choice1
```

配置文件遵循一些惯例（简洁起见，排除了诸如 `USER_NAME` 之类的自解释惯例）：

`COMPUTER_NAME`：指示服务器的有效名称（例如 `localhost`、`server.cepremap.org`）或 IP 地址。

`DRIVE_NAME`：在 Windows 中指示一个有效的驱动器名，没有尾随冒号（例如 `C`）。

`PATH`：指示底层操作系统中的有效路径（例如`/home/user/dynare/matlab/`）。

`PATH_AND_FILE`：指示到底层操作系统中的文件的有效路径（例如`/usr/local/MATLAB/R2010b/bin/matlab`）。

`BOOLEAN`：是 `true` 还是 `false`

5.1 Dynare 配置

本节说明如何配置 Dynare 用于一般处理。目前，只有一个选项可用。

`Configuration block:[hooks]`

指定运行 Dynare 时使用的配置选项。

选项

`GlobalInitFile=PATH_AND_FILE`：全局初始化文件的位置，会运行在 `global_initialization.m` 的结束位置。

示例

```
[hooks]
GlobalInitFile=/home/usern/dynare/myInitFile.m
```

Configuration block:[paths]

指定运行 Dynare 时将使用的路径。

选项

`Include=PATH`

在通过`@#include` 搜索要包含的文件时使的冒号分隔路径。通过`-I` 优先于这里指定的路径，而这些路径优先于`@#include` 指定的路径。

示例

```
[paths]
Include=/path/to/folder/containing/modfiles:/path/to/another
/folder
```

5.2 平行配置

本节将解释如何配置 Dynare，以并行化一些只需要很少进程间通信的任务。

并行化是通过在本地或远程机器上运行多个 MATLAB 或 Octave 来完成的。主进程和从进程之间的通信是通过 Windows 上的 SMB 和 UNIX 上的 SSH 来完成的。输入和输出数据以及一些简短的状态消息通过网络文件系统交换。目前，系统只能使同构网格：只有 Windows 或 Unix 计算机。

以下例程目前是并行的：

- 多链检测时的后验抽样算法；
- Metropolis-Hastings 诊断；
- 后验 IRF；
- 事前及事后统计；
- 一些绘图程序；

注意，为了触发计算的并行化，仅仅创建配置文件是不够的：您还需要为 `dynare` 命令指定 `parallel` 选项。有关并行化引擎的更多细节和其他选项，参见 3.1 Dynare 调用。

您还需要验证您的集群（由一个主节点和一个或多个从节点组成）是否满足以下要求：

对于 Windows 网格：

- 必须有一个标准的 Windows 网络（SMB）；
- PsTools 必须安装在主 Windows 机器的路径中；
- 主机器上的 Windows 用户必须是集群中任何其他从机器的用户，该用户将用于远程计算；
- 详细安装说明可以在 5.3 Windows 分布指南找到。

对于 UNIX 网格：

- SSH 必须安装在主机器和从机器上；
- 必须安装 SSH 密钥，以便在不使用密码的情况下完成从主服务器到从服务器的 S

SH 连接，或者使用 SSH 代理。

警告：主从之间的兼容性考虑，强烈建议在主服务器和所有从服务器上使用相同版本的 Dynare。不同的版本通常会导致一些问题，比如评估期间的接受率为零。当升级到一个新的 Dynare 版本时，不要忘记调整 DynarePath。

我们现在转向配置指令描述。注意，配置文件中注释可由 hashtag(#)开头的单独行提供。

Configuration block:[cluster]

在并行工作时，需要`[cluster]`指定要使用的计算机组。即使您只在一台计算机上调用多个进程，也需要它。

选项

Name=CLUSTER_NAME：该集群的引用名称。

Members=NODE_NAME[(WEIGHT)]NODE_NAME[(WEIGHT)]...：包含集群的节点列表，其中包含为该节点指定的可选计算权值。计算权值表示一个节点相对于其他节点的能力有多强（例如 `n1（2）n2(1)n3（3）`，表示 `n1` 比 `n2` 强大 `2` 倍而 `n3` 比 `n2` 强大 `3` 倍）。每个节点至少由一个空格隔开，权重在括号中，没有空格将它们与节点分开。

示例

```
[cluster]
Name=c1
Members=n1 n2 n3


[cluster]
Name=c2
Members=n1(4) n2 n3
```

Configuration block:[node]

当并行工作时，将使的台计算机都需要`[node]`。所需的选项各不相同，这取决于底层操作系统以及您是在本地工作还是远程工作。

选项

Name=NODE_NAME：节点的引用名称

CPUnbr=INTEGER|[INTEGER:INTEGER]：如果只传递一个整数，则使用处理器数量。如果传递了一个整数范围，则使用的特定处理器计数定义为从 1 开始，而不是从 0 开始）。注意，只有在 Windows 下才能使特定的处理器；在 Linux 和 macOS 中，如果传递了一个范围，将使用相同数量的处理器，但是范围将被调整为从一个开端开始。

ComputerName=COMPUTER_NAME：节点的名称或 IP 地址。如果想在本地运行，请使用 `locallost`（区分大小写）。

Port=INTEGER：节点上要连接的端口号。默认为空，这意味着连接将被连接到默认

的 SSH 端口（22）。

Username=USER_NMAE：用于登录远程系统的用户名。需要在所有平台上远程运行。

Password=PASSWORD：用于登录远程系统的密码。需要源自 Windows 远程运行。

RemoteDrive=DRIVE_NAME：用于远程计算的驱动器。需要源自 Windows 远程运行。

RemoteDirectory=PATH：用于远程计算的路径。需要在所有平台上远程运行。

DynarePath=PATH：在 Dynare 安装目录中到 matlab 子目录的路径。默认值是空字符串。

MatlabOctavePath=PATH_AND_FILE：到 MATLAB 或 Octave 可执行文件的路径。默认值是 `matlab`。

NumberOfThreadsPerJob=INTEGER：对于 Windows 节点，设置分配给每个远程 MATLAB 或 Octave 的线程数。默认值是 `1`。

SingleCompThread=BOOLEAN：是否禁 MATLAB 的本机多线程。默认值为 `false`。在 Octave 下没有意义的选项。

OperatingSystem=OPEMTING_SYSTEM：与节点相关联的操作系统。只有在使用来自不同操作系统的节点创建集群时才需要。可能的值是 `unix` 或 `windows`。没有默认值。

示例

```
[node]
Name=n1
ComputerName=localhost
CPUnbr=1

[node]
Name=n2
ComputerName=dynserv.cepremap.org
CPUnbr=5
UserName=usern
RemoteDirectory=/home/usern/Remote
DynarePath=/home/usern/dynare/matlab
MatlabOctavePath=matlab

[node]
Name=n3
ComputerName=dynserv.dynare.org
Port=3333
```

```
CPUnbr=[2:4]
UserName=usern
RemoteDirectory=/home/usern/Remote
DynarePath=/home/usern/dynare/matlab
MatlabOctavePath=matlab
```

5.3 Windows 分步指南

本节概述了在大多数 Windows 系统上为并行执行建立 Dynare 所必需的步骤。

1．编写包含所需选项的配置文件。下面是一个最小的工作示例，它设置了一个由两个本地 CPU 内核组成的集群，允许并行运行两个 MCMC 链；

2．保存配置文件到某处。如果您为它提供了 `conffile` 命令行选项，那么名称和文件结尾并不重要。唯一的约束是路径必须是有效的文件名，不包含非字母数字字符，也不包含任何空格。要访问配置文件而不在命令行提供显式路径，必须将其保存在名称 `dynare.ini` 下进入您的户账户的 `Appliation Data` 文件夹；

3．从 https://technet.microsoft.com/sysinternals/pstools.aspx 安装 PSTools 进入您的系统的，例如进入 `C:\PSTools`；

4．将 Windows 系统路径设置为 `PSTools` 文件夹（例如，使类似于按下 Windows 键+Pause 这样的方法来打开系统配置，然后转到高级->环境变量->路径）；

5．重新启动计算机以使路径更改生效；

6．打开 Matlab，在命令窗口输入：

```
!psexec
```

这将从 PSTools 执行您的系统上的 `psexec.exe`。并且显示 Dynare 是否能定位它。如果 Matlab 在此阶段发出抗议，说明您没有正确设置 PSTools 文件夹的 Windows 系统路径；

7．如果 `psexec.exe` 位于前面的步骤中，弹出窗口将出现，要求确认许可协议。确认 `psexec` 的版权声明（只需要做一次）。在此之后，Dynare 应该准备好并行执行；

8．调用`*.mod` 文件上的 Dynare，调用 `parallel` 选项并使用 `conffile` 选项提供配置文件的路径（如果您没有在步骤 2 中将它保存为`%APPDATA%\Dynare.ini` 的话，它应该被自动检测）；

```
dynare ls2003 parallel conffile='C:\Users\Dynare~1\parallel\
conf_file.ini'
```

注意，在 `conffile` 路径中不允许出现超过 8 个字符的空白或名称。8 个字符的约束可以通过使用波浪窗口路径标记来实现，如上例所示。

示例

```
#cluster needs to always be defined first
[cluster]
```

```
#Provide a name for the cluster
Name=Local
#declare the nodes being member of the cluster
Members=n1

#declare nodes (they need not all be part of a cluster)
[node]
#name of the node
Name=n1
#name of the computer (localhost for the current machine)
ComputerName=localhost
#cores to be included from this node
CPUnbr=[1:2]
#path to matlab.exe; on Windows,the MATLAB bin folder is in
the system path
#so we only need to provide the name of the exe file
MatlabOctavePath=matlab
#Dynare path you are using
DynarePath=C:/dynare/4.7.0/matlab
```

6 时间序列

Dynare 提供了一个于处理时间序列数据的 Matlab/Octave 类，还提供了新的日期类型，这样用户不必担心日期的类型和方法。下面，您将首先找到于创建和处理日期的类型和方法，然后找到用于操作时间序列的类。Dynare 还提供了 X-13 ARIMA-SEATS 季节性调整程序的接口，该程序由美国人口普查局（the US Census Bureau，2017）制作、发行和维护。

6.1 日期

6.1.1 mod 文件中的日期

Dynare 理解 mod 文件中的日期。用户用以下语法声明年度、半年度、季度或月度：

```
1990Y
1990S2
1990Q4
1990M11
```

幕后的 Dynare 预处理器将这些表达式翻译成下文描述的 MATLAB/Octave 的 `dates` 类实例。基本操作可以按期执行：

加二进制运算符（+）：一个整数标量，解释为若干周期，可以添加到一个日期。例如，如果 `a=1950Q1`，那么 `b=1951Q2` 和 `b=a+5` 是相同的。

加一元运算符（+）：将日期增加一个周期。`+1950Q1` 与 `1950Q2` 相同，`++++1950Q1` 与 `1951Q1` 相同。

减二进制运算符（-）：有两种功能：差分和差集。如果第二个参数是日期，则第一个日期和第二个日期之间的差值（例如 `1951Q2-1950Q1` 等于 5）。如果第二个参数是整数 `X`，则从日期里减去 `X` 个周期（例如 `1951Q2-2` 等于 `1950Q4`）。

减一元运算符（-）：将一个周期减为一个日期。`-1951Q1` 和 `1949Q4` 是一样的。一元减号运算符是一元加号运算符的逆运算，`+-1950Q1` 和 `1950Q1` 是一样的。

冒号运算符（:）：可用于创建一系列日期。例如，`r=1950Q1:1951Q1` 创建了一个包含五个元素的日期对象：`1950Q1`、`1950Q2`、`1950Q3`、`1950Q4` 和 `1951Q1`。默认相邻元素的增量是一个周期。可以使用以下指令来更改这个默认值：`1950Q1:2:1951Q1`，它将实例化一个含三个元素的 `dates` 对象：`1950Q1`、`1950Q3` 和 `1951Q1`。

horzcat 运算符（[,]）：连接日期对象而不删除重复内容。例如`[1950Q1,1950Q2]`是一个带有两个元素（`1950Q1` 和 `1950Q2`）的日期对象。

vertcat 算子（[;]）：和 `horzcat` 操作符一样。

eq 算子（equal，==）：测试两个 `dates` 对象是否相等。`+1950Q1==1950Q2` 返回 `true`，`1950Q1==1950Q2` 返回 `false`。如果比较对象都有 `n>1` 元素，那么 `eq` 操作符返回一个 `n×1` 的逻辑列向量。

ne 算子（not equal，~=）：测试两个 date 对象是否不等。+1950Q1~=返回 false，而 1950Q1~=1950Q2 返回 true。如果比较对象都有 n>1 个元素，则运算符返回一个 n×1 的逻辑列向量。

lt 操作符（less than，<）：测试 dates 对象是否早于另一个 dates 对象。例如，1950Q1<1950Q3 返回 true。如果比较对象同时具有 n>1 个的元素，则 It 操作符返回一个 n×1 的逻辑列向量。

gt 运算符（greater than，>）：测试 dates 对象是否晚于另一个 dates 对象。例如，1950Q1>1950Q3 返回 false。如果比较对象同时具有 n>1 个的元素。gt 运算符返回一个 n×1 的逻辑列向量。

le 运算符（less or equal，<=）：测试 dates 对象是否不晚于另一个 dates 对象。例如，1950Q1<=1950Q3 返回 true。如果比较对象同时具有 n>1 个的元素，那么运算符返回一个 n×1 的逻辑列向量。

ge 运算符（greater or equal，>=）：测试 dates 对象是否不早于另一个 dates 对象。例如，1950Q1>=1950Q3 返回 false。如果比较对象同时具有 n>1 个的元素，则 ge 运算符返回一个 n×1 的逻辑列向量。

人们可以在 dates 对象中选择一个或一些元素，就像他会从 MATLAB/Octave 中的向量中提取一些元素一样。a=1950Q1:1951Q1 成为 dates 对象，然后 a(1)==1950Q1 返回 true，a(end)==1951Q1 返回 true 并且 a(end-1:end)选择 a 的最后两个元素（通过实例化日期对象[1950Q4,1951Q1]）。

备注：Dynare 将*.mod 文件中出现的任何日期替换为日期类的实例，而不管上下文如何。例如，d=1950Q1 将被翻译为 d=date('1950Q1');。如果在字符串中定义了日期，自动替换可能会导致崩溃。通常，如果用户想要显示日期：

```
disp('Initial period is 1950Q1');
```

Dynare 将翻译为：

```
disp('Initial period is dates('1950Q1')');
```

这将导致崩溃，因为此表达式在 MATLAB 中是非法的。对于这种情况，Dynare 提供了逃逸参数$。以下表达式：

```
disp('Initial period is $1950Q1');
```

将会在生成的 MATLAB 脚本中翻译成：

```
disp('Initial period is 1950Q1');
```

6.1.2 日期类

Dynare class:dates

成员

- **freq**：等于 1、2、4、12 或 365（代表年度、半年度、季度、月度或日期）；

- **time**：一个 n*1 数组，年份 0（）以来的期数。

每个成员都是单独的，可以显示成员的内容，但不能直接更改数值。注意，不可能在 dates 对象中混合不同频率，所有的元素必须有同频。

dates 类具有以下构造函数：

Constructor:dates()

Constructor:dates(FREQ)

返回具有给定频率的空的日期对象（如果使用一个输入参数调用构造函数）。FREQ 是一个等于“Y”或“A”（年）、“S”或“H”（半年）、“Q”（季）、“M”（月）或“D”（日）的字符。注意，FREQ 不区分大小写，因此，例如，“q”也允许定义季度。频率也可以用等于 1（年）、2（半年）、4（季）、12（月）或 365（日）的整数标量设置。空对象的实例化可用于重命名 dates 类。例如，如果只处理季度，对象 qq 可以创建为：

```
qq=dates('Q')
```

和一个 dates 对象持有日期 2009Q2：

```
d0=qq(2009,2);
```

如果 dates 对象必须以编程方式定义，那就简单多了。对于具体日期，我们可以实例化一个空的具体日期对象为

```
dd=dates('D')
```

以及 date 对象包含日期 2020-12-31

```
d1=dd(2020,12,31);
```

Constructor:dates(STRING)

Constructor:dates(STRING,STRING,...)

返回一个表示由字符串 STRING 给出的日期的 dates 对象。该字符串必须可解释为日期（只有以下形式的字符串才允许：'1990Y'、'1990A'、'1990Q1'、'1990M2'或'2020-12-31'），例行程序 isdate 可用于测试字符串是否可解释为日期。如果提供了多个参数，它们都应该是表示为字符串的日期，结果的 dates 对象包含与构造函数参数一样多的元素。对于具体日期，字符串必须是 yyyy-mm-dd 的形式，其中月（mm）和日（dd）为两位数，即使日或月的数值小于 10（十位上用 0 占位）。

Constructor:dates(DATES)

Constructor:dates(DATES,DATES,...)

返回作为输入参数传递的 dates 对象的副本。如果提供了多个参数，它们都应该是 dates 对象。实例化的 dates 对象中的元素数量等于作为参数传递给构造函数的 dates 中的元素之和。

Constructor:dates(FREQ,YEAR,SUBPERIOD[,S])

其中 FREQ 是单个字符（'Y'、'A'、'S'、'H'、'Q'、'M'、'D'）或指定频率的

整数（1、2、4、12 或 365）。YEAR、SUBPERIOD 和 S 是 n*1 个整数的向量。返回一个包含 n 个元素的 dates 对象。最后一个参数 S 仅适用于单日频率。如果 FREQ 等于'Y'、'A'或 1，则不需要第三个参数（因为此时 SUBPERIOD 必须是 1 向量）。

示例

```
do1=dates('1950Q1');
do2=dates('1950Q2','1950Q3');
do3=dates(do1,do2);
do4=dates('Q',1950,1);
do5=dates('D',1973,1,25);
```

下面列出了按字母顺序排列的可用方法。注意，默认情况下，这些方法不允许就地修改，当一个方法应用到一个对象时，一个新对象会被实例化。例如，要将方法 multiply bytwo 应用于对象 X，我们可以这样写：

```
>>X=2;
>>Y=X.multiplybytwo();
>>X

2

>>Y

4
```

或者等价形式：

```
>>Y=multiplybytwo(X);
```

对象 X 保持不变，对象 Y 是 X 的修改副本（乘以 2）。如果方法名称后面加下划线，则变更此行为。此时可以避免创建副本。例如，按照前面的例子，我们将有：

```
>>X=2;
>>X.multiplybytwo_();
>>X

4
```

如果在循环中调用这些方法，在适当位置使用下划线方法修改对象特别有用，因为节省了对象实例化的支出。

Method:C=append(A,B)

Method:C=append_(A,B)

将 dates 对象 B，或一个可以解释为日期的字符串附加到 dates 对象 A。如果 B 是一个 dates 对象，则假定至多有一个元素。

示例

```
>>D=dates('1950Q1','1950Q2');
>>d=dates('1950Q3');
>>E=D.append(d);
>>F=D.append('1950Q3');
>>isequal(E,F)

ans =

     1
>>F

F=<dates:1950Q1,1950Q2,1950Q3>

>>D

D=<dates:1950Q1,1950Q2>

>>D.append_('1950Q3')

ans=<dates:1950Q1,1950Q2,1950Q3>
```

Method:B=char(A)

重载 MATLAB/Octave 的 char 函数。dates 对象转换为字符数组。

示例

```
>>A=dates('1950Q1');
>A.char()

ans=

'1950Q1'
```

Method:C=colon(A,B)

Method:C=colon(A,i,B)

重载 MATLAB/Octave 的冒号(:)操作符。A 和 B 是 dates 对象。可选的增量 i 是一个标量整数（默认值为 i=1）。此方法返回一个 dates 对象，并用于创建日期范围。

示例

```
>>A=dates('1950Q1');
>>B=dates('1951Q2');
>>C=A:B

C=<dates:1950Q1,1950Q2,1950Q3,1950Q4,1951Q1>

>>D=A:2:B

D=<dates:1950Q1,1950Q3,1951Q1>
```

Method:B=copy(A)

返回 dates 对象的副本。

Method:disp(A)

重载 MATLAB/Octave 的 disp 函数的 dates 对象。

Method:display(A)

重载 MATLAB/Octave 的 display 函数的 dates 对象。

示例

```
>>disp(B)

B=<dates:1950Q1,1950Q2,1950Q3,1950Q4,1951Q1,1951Q2,1951Q3,19
51Q4,1952Q1,1952Q2,1952Q3>



>>display(B)

B=<dates:1950Q1,1950Q2,...,1952Q2,1952Q3>
```

Method:B=double(A)

重载 MATLAB/Octave 的 double 函数。A 是 dates 对象。该方法返回一个 dates 对象的浮点表示，整数和小数部分分别对应年份和子期间。小数部分是子期间的编号减 1 除以频率（1、4 或 12）

示例

```
>>a=dates('1950Q1'):dates('1950Q4');
```

```
>>a.double()

ans =

     1950.00
     1950.25
     1950.50
     1950.75
```

Method:C=eq(A,B)

重载 MATLAB/Octave 的 eq（等号，==）运算符。dates 对象 A 和 B 必须具相同数量的元素（例如 n)。返回的参数是 n×1 的逻辑向量。当且仅当日期 A(i)和 B(i)相同时，C 的第*i*个元素等于 true。

示例

```
>>A=dates('1950Q1','1951Q2');
>>B=dates('1950Q1','1950Q2');
>>A==B

ans =

  2x1 logical array

   1
   0
```

Method:C=ge(A,B)

重载 MATLAB/Octave 的 ge（不小于，>=）运算符。dates 对象 A 和 B 必须具有相同数量的元素（例如 n)。返回的参数是 n×1 的逻辑向量。当且仅当日期 A(i)是不早于 B(i)时，C 的第 i 个元素等于 true。

示例

```
>>A=dates('1950Q1','1951Q2');
>>B=dates('1950Q1','1950Q2');
>>A>=B

ans =
```

```
  2x1 logical array

   1
   1
```

Method:C=gt(A,B)

重载 MATLAB/Octave 的 gt（大于，>）运算符。dates 对象 A 和 B 必须有相同数量的元素（例如 n)。返回的参数是 n×1 的逻辑向量。当且仅当日期 A(i)晚于日期 B(i)时，C 的第 i 个元素等于 1。

示例

```
>>A=dates('1950Q1','1951Q2');
>>B=dates('1950Q1','1950Q2');
>>A>B

ans =

  2x1 logical array

   0
   1
```

Method:D=horzcat(A,B,C,...)

重载 MATLAB/Octave 的 horzcat 操作符。所有输入参数必须是 dates 对象。返回的参数是一个在输入参数中收集了所有日期的 dates 对象（不删除重复)。

示例

```
>>A=dates('1950Q1');
>>B=dates('1950Q2');
>>C=[A,B];
>>C

C=<dates:1950Q1,1950Q2>
```

Method:C=intersect(A,B)

重载 MATLAB/Octave 的 intersect 函数。所有输入参数必须是 dates 对象。返回的参数是一个在输入参数中收集了所有常见日期的 dates 对象。如果 A 和 B 是不相交的 dates 对象，返回空 dates 对象。在 dates 对象 C 返回的日期按递增顺序排序。

示例

```
>>A=dates('1950Q1'):dates('1951Q4');
>>B=dates('1951Q1'):dates('1951Q4');
>>C=intersect(A,B);
>>C


C=<dates:1951Q1,1951Q2,1951Q3,1951Q4>
```

Method:B=isempty(A)

重载 MATLAB/Octave 的 isempty 函数。

示例

```
>>A=dates('1950Q1');
>>A.isempty()


ans =


  logical


  0


>>B=dates();
>>B.isempty()


ans =


  logical


  1
```

Method:C=isequal(A,B)

重载 MATLAB/Octave 的 isequal 函数。

示例

```
>>A=dates('1950Q1');
>>B=dates('1950Q2');
>>isequal(A,B)


ans =
```

```
  logical

  0
```

Method:C=le(A,B)

重载 MATLAB/Octave 的 le（不大于，<=）运算符。dates 对象 A 和 B 必须具有相同数量的元素（例如 n)。返回的参数是 n×1 的逻辑向量。当且仅当日期 A(i)不晚于日期 B(i)时，C 的第 i 个元素等于 true。

示例

```
>>A=dates('1950Q1','1951Q2');
>>B=dates('1950Q1','1950Q2');
>>A<=B

ans =

  2x1 logical array

   1
   0
```

Method:B=length(A)

重载 MATLAB/Octave 的长度函数。返回 dates 对象中的元素数量。

示例

```
>>A=dates('1950Q1'):dates(2000Q3);
>>A.length()

ans =

   203
```

Method:C=lt(A,B)

重载 MATLAB/Octave 的 lt（小于，<）运算符。dates 对象 A 和 B 必须具有相同数量的元素（例如 n)。返回的参数是 n×1 的逻辑向量。当且仅当日期 A(i)早于日期 B(i)时，C 的第*i*个元素等于 true。

示例

```
>>A=dates('1950Q1','1951Q2');
```

```
>>B=dates('1950Q1','1950Q2');
>>A<B


ans =


  2x1 logical array


   0
   0
```

Method:D=max(A,B,C,...)

重载 MATLAB/Octave 的 max 函数。所有输入参数必须是 dates 对象。函数返回包含最大日期的单个元素 dates 对象。

示例

```
>>A={dates('1950Q2'),dates('1953Q4','1876Q2'),dates('1794Q3
')};
>>max(A{:})


ans=<dates:1953Q4>
```

Method:D=min(A,B,C,...)

重载 MATLAB/Octave 的 min 函数。所有输入参数必须是 dates 对象。函数返回包含最小日期的单个元素 dates 对象。

示例

```
>>A={dates('1950Q2'),dates('1953Q4','1876Q2'),dates('1794Q3
')};
>>min(A{:})


ans=<dates:1794Q3>
```

Method:C=minus(A,B)

重载 MATLAB/Octave 的 minus 运算符（-）。如果两个输入参数都是 dates 对象，然后返回 A 和 B 之间的周期数（A+C=B）。如果 B 是整数向量，减号运算符将 dates 对象向后移动 B 个周期。

示例

```
>>d1=dates('1950Q1','1950Q2','1960Q1');
>>d2=dates('1950Q3','1950Q4','1960Q1');
```

```
>>ee=d2-d1

ee =

     2
     2
     0

>>d1-(-ee)

ans=<dates:1950Q3,1950Q4,1960Q1>
```

Method:C=mtimes(A,B)

重载 MATLAB/Octave 的 mtimes（*）运算符。A 和 B 分别为 dseries 对象和标量整数。返回对象 A 复制 B 次的 dates。

示例

```
>>d=dates('1950Q1');
>>d*2

ans=<dates:1950Q1,1950Q1>
```

Method:C=ne(A,B)

重载 MATLAB/Octave 的 ne（不等于，~=）运算符。dates 对象 A 和 B 必须具有相同数量的元素（例如 n），或其中一个输入必须是单个元素 dates 对象。返回的参数是 n×1 的逻辑向量。当且仅当 datesA（i）和 B（i）不同时，C 的第*i*个元素等于 true。

示例

```
>>A=dates('1950Q1','1951Q2');
>>B=dates('1950Q1','1950Q2');
>>A~=B

ans =

  2x1 logical array

   0
   1
```

Method:C=plus(A,B)

重载 MATLAB/Octave 的 plus 运算符（+）。如果两个输入参数都是 dates 对象，然后该方法结合 A 和 B 而不删除重复的。如果 B 是整数向量，plus 运算符将 dates 对象向前移动 B 个周期。

示例

```
>>d1=dates('1950Q1','1950Q2')+dates('1960Q1');
>>d2=(dates('1950Q1','1950Q2')+2)+dates('1960Q1');
>>ee=d2-d1;

ee =

     2
     2
     0

>>d1+ee
ans=<dates:1950Q3,1950Q4,1960Q1>
```

Method:C=pop(A)

Method:C=pop(A,B)

Method:C=pop_(A)

Method:C=pop_(A,B)

dates 类的 pop 方法。如果仅提供一个输入，则删除最后一个元素 dates 对象。如果提供了第二个输入参数，标量整数介于 1 和 A.length()，从 dates 对象 A 中删除元素编号 B。

示例

```
>>d=dates('1950Q1','1950Q2');
>>d.pop()

ans=<dates:1950Q1>

>>d.pop_(1)

ans=<dates:1950Q2>
```

Method:C=remove(A,B)

Method:C=remove_(A,B)

删除 dates 类的方法。两个输入都必须是 dates 对象，从 A 删除 B 的日期。

示例

```
>>d=dates('1950Q1','1950Q2');
>>d.remove(dates('1950Q2'))


ans=<dates:1950Q1>
```

Method:C=setdiff(A,B)

重载 MATLAB/Octave 的 setdiff 函数。所有的输入参数必须是 dates 对象。返回的参数个 dates 对象，所有的 dates 都出现在 A 中，而不是在 B 中。如果 A 和 B 是不相交的 dates 对象，函数返回 A。dates 对象 C 的返回日期递增排序。

示例

```
>>A=dates('1950Q1'):dates('1969Q4');
>>B=dates('1960Q1'):dates('1969Q4');
>>C=dates('1970Q1'):dates('1979Q4');
>>setdiff(A,B)


ans=<dates:1950Q1,1950Q2,...,1959Q3,1959Q4>


>>setdiff(A,C)


ans=<dates:1950Q1,1950Q2,...,1969Q3,1969Q4>
```

Method:B=sort(A)

Method:B=sort_(A)

dates 对象的排序方法。返回一个元素递增排序的 dates 对象。

示例

```
>>dd=dates('1945Q3','1938Q4','1789Q3');
>>dd.sort()


ans=<dates:1789Q3,1938Q4,1945Q3>
```

Method:B=strings(A)

将 dates 对象转换为字符数组单元。

示例

```
>>A=dates('1950Q1');
```

```
>>A=A:A+1;
>>A.strings()

  ans =

    1x2 cell array

    {'1950Q1'}    {'1950Q2'}
```

Method:B=subperiod(A)

返回日期的子周期（1 和 A.freq 之间的整数标量），该方法不适用于每日频率。

示例

```
>>A=dates('1950Q2');
>>A.subperiod()

ans =

      2
```

Method:B=uminus(A)

重载 MATLAB/Octave 的一元减法运算符，返回一个元素向后移动一位的 dates 对象。

示例

```
>>dd=dates('1945Q3','1938Q4','1973Q1');
>>-dd

ans=<dates:1945Q2,1938Q3,1972Q4>
```

Method:D=union(A,B,C,...)

重载 MATLAB/Octave 的 union 函数。返回元素递增排序的 dates 对象（删除重复，为了保持重复，使用 horzcat 或 plus 运算符）

示例

```
>>d1=dates('1945Q3','1973Q1','1938Q4');
>>d2=dates('1973Q1','1976Q1');
>>union(d1,d2)

ans=<dates:1938Q4,1945Q3,1973Q1,1976Q1>
```

Method:B=unique(A)

Method:B=unique_(A)

重载 MATLAB/Octave 的 unique 函数程序。返回 dates 对象的同时删除重复（只保留最后一次出现的日期）

示例

```
>>d1=dates('1945Q3','1973Q1','1945Q3');
>>d1.unique()

ans=<dates:1973Q1,1945Q3>
```

Method:B=uplus(A)

重载 MATLAB/Octave 的单数加号运算符。返回领先一个周期元素的 dates 对象。

示例

```
>>dd=dates('1945Q3','1938Q4','1973Q1');
>>+dd

ans=<dates:1945Q4,1939Q1,1973Q2>
```

Method:D=vertcat(A,B,C,...)

重载 MATLAB/Octave 的 horzcat 算子。所有的输入参数必须是 dates 对象。返回的参数是一个收集输入参数中给定所有日期的 dates 对象（不删除重复）。

Method:B=year(A)

返回日期的年份（1 和 A.freq 之间的整数标量）。

示例

```
>>A=dates('1950Q2');
>>A.subperiod()

ans =

      1950
```

6.2 dseries 类

Dynare class:dseries

MATLAB/Octave 的 dseries 类用来处理时间序列数据。像任何 MATLAB/Octave 语句一样，这个类可以在 Dynare 的 mod 文件中使用。一个 dseries 对象有六个成员：

成员

- **name**：一个 vobs*1 的字符串数组或 vobs*p 字符数组，即变量的名称；
- **tex**：一个 vobs*1 的字符串数组或 vobs*p 字符数组，即变量的在 tex 文本下

的名称；

- **dates**(*dates*)：一个带有 nobs 元素的对象，即样本的日期；
- **data**(*double*)：一个 nobs×vobs 数组，即数据；
- **ops**：在变量上操作的历史；
- **tags**：用户定义在变量上的标记。

data、name、tex 和 ops 系用户自定义成员。下面是可用的构造函数：

Constructor:dseries()

Constructor:dseries(INITIAL_DATE)

实例化一个空的 dseries 对象，如果已定义，则使用单个元素 dates 对象 *INITIAL_DATE* 给出的初始日期。

Constructor:dseries(FILENAME[,INITIAL_DATE])

输入字符串 *FILENAME* 传递该名称文件定义的数据用来实例化并填充一个 dseries 对象。有效的文件类型是*.m、*.mat、*.csv 和*.xls/*.xlsx（Octave 只支持*.xlsx 文件，且必须安装 Octave-forge 的 io 包）。应该明确提供文件的扩展名。一个典型的*.m 文件有以下形式：

```
FREQ__=4;
INIT__='1994Q3';
NAMES__={'azert';'yuiop'};
TEX__={'azert';'yuiop'};

azert=randn(100,1);
yuiop=randn(100,1);
```

如果改用*.mat 文件，它应该提供相同的信息，但是数据不应由一组向量给出，而应该由一个命名为 DATA__的双精度矩阵给出。该矩阵的列数应与 NAMES__中元素的个数（变量个数）一样多。注意，INIT__变量可以是 dates 对象，也可以是可用于实例化相同 dates 对象的字符串。如果在*.mat 或*.m 文件中未提供 INIT__，则初始值默认被设置为 date('1Y')。如果将第二个参数 dates 对象 *INITIAL_DATE* 传递给构造函数，则 *FILENAME* 定义的初始日期将被重置为 *INITIAL_DATE*。这在数据文件未提供 INIT__情况时很有用。

Constructor:dseries(DATA_MATRIX[,INITIAL_DATE[,LIST_OF_NAMES[,TEX_NAMES]]])

Constructor:dseries(DATA_MATRIX[,RANGE_OF_DATES[,LIST_OF_NAMES[,TEX_NAMES]]])

如果数据不是从文件中读取的，则可以通过$T \times N$矩阵作为第一个实参提供给 dserie

s 的构造函数，其中T表示对N个变量的观测数。可选的第二个参数 *INITIAL_DATE* 可以是表示观测初始时期的 dates 对象，也可以是用于实例化 dates 对象的字符串。它的默认值是 dates('1Y')。可选的第三个参数 *LIST_OF_NAMES* 是一个$N \times 1$的字符串数组，每个变量名称都有一个条目。与 *DATA_MATRIX* 第 i 列关联的默认名称是 Variable_i。最后一个参数 *TEX_NAMES* 是与变量关联的 LaTeX 名称组成的$N \times 1$字符串数组。与 *DATA_MATRIX* 的第 i 列关联的默认 LaTeX 名称是 Variable_i。如果可选的第二个参数指定的是日期范围，即 dates 对象如果是 *RANGE_OF_DATES*，则第一个参数中矩阵的行数必须与 *RANGE_OF_DATES* 中元素的个数相符或等于 1（在这种情况下，将复制单个观测值）。

Constructor:dseries(TABLE)

在 MATLAB 表作为单一参数的情况下，创建一个 dseries 对象。此处，应给定表的第一列包含 dseries 的日期，表的第一行包含名称。此功能在 Octave 或 MATLAB R2013a 以更早的版本中不可用。

示例

创建一个 dseries 对象的各种方法：

```
do1=dseries(1999Q3);
do2=dseries('filename.csv');
do3=dseries([1;2;3],1999Q3,{'var123'},{'var_{123}'});


>>do1=dseries(dates('1999Q3'));
>>do2=dseries('filename.csv');
>>do3=dseries([1;2;3],dates('1999Q3'),{'var123'},{'var_{123}'});
```

我们可以使用重载小括号符从 dseries 对象轻松地创建子样本。如果 ds 是具有T个观测值的 dseries 对象，而d是具有$S < T$元素的 dates 对象，则在$min(d)$不小于 ds 中的第一个观测值关联的日期，并且$max(d)$不大于与最后一个观测值相关联的日期的情况下，ds(d)实例化一个新的 dseries 对象，其中包含了由 d 定义的子样本。

下面按字母顺序列出了可用的方法。与上一节一样，方法的修改版本在适当的位置以下划线作为后缀。

Method:A=abs(B)

Method:abs_(B)

重载 dseries 对象的 abs()函数。返回在 dseries 对象 B 中变量的绝对值。

示例

```
>>ts0=dseries(randn(3,2),'1973Q1',{'A1';'A2'},{'A_1';'A_2'});
>>ts1=ts0.abs();
>>ts0
```

```
ts0 is a dseries object:

      |A1      |A2
1973Q1|-0.67284|1.4367
1973Q2|-0.51222|-0.4948
1973Q3|0.99791 |0.22677

>>ts1

ts1 is a dseries object:

      |abs(A1)|abs(A2)
1973Q1|0.67284|1.4367
1973Q2|0.51222|0.4948
1973Q3|0.99791|0.22677
```

Method:[A,B]=align(A,B)

Method:align_(A,B)

如果在不同的时间范围内定义 dseries 对象 A 和 B，则此函数使用 NaN 在 A 和/或 B 空缺位置上占位，以使得它们被定义在相同的时间范围内。注意，两个 dseries 对象必须有相同的频率。

示例

```
>>ts0=dseries(rand(5,1),dates('2000Q1'));%2000Q1->2001Q1
>>ts1=dseries(rand(3,1),dates('2000Q4'));%2000Q4->2001Q2
>>[ts0,ts1]=align(ts0,ts1);        %2000Q1->2001Q2
>>ts0

ts0 is a dseries object:

      |Variable_1
2000Q1|0.81472
2000Q2|0.90579
2000Q3|0.12699
2000Q4|0.91338
```

```
2001Q1|0.63236
2001Q2|NaN

>>ts1

ts1 is a dseries object:

      |Variable_1
2000Q1|NaN
2000Q2|NaN
2000Q3|NaN
2000Q4|0.66653
2001Q1|0.17813
2001Q2|0.12801

>>ts0=dseries(rand(5,1),dates('2000Q1'));%2000Q1->2001Q1
>>ts1=dseries(rand(3,1),dates('2000Q4'));%2000Q4->2001Q2
>>align_(ts0,ts1); %2000Q1->2001Q2
>>ts1

ts1 is a dseries object:

      |Variable_1
2000Q1|NaN
2000Q2|NaN
2000Q3|NaN
2000Q4|0.66653
2001Q1|0.17813
2001Q2|0.12801
```

Method:C=backcast(A,B[,diff])

Method:backcast_(A,B[,diff])

使用 dseries 对象 B 的增长率后向推算 dseries 对象 A（但若最后一个可选参数 diff 为真，则后向推算将使用一阶差分）。两个 dseries 对象必须有相同的频率。

Method:B=baxter_king_filter(A,hf,lf,K)

Method:baxter_king_filter_(A,hf,lf,K)

对 dseries 对象实施 Baxter 和 King（1999）带通滤波。该滤波使用具有$2K+1$个点的对称移动平均平滑器，分离出周期长度介于 hf（高频）到 lf（低频）之间的商业周期波动，以便在样本的开头和结尾处的K个观测值在过滤器的计算中剔除。hf 的默认值为 6，lf 的默认值为 32，K 的默认值为 12。

示例

```
%Simulate a component model (stochastic trend,deterministic
%trend, and a stationary autoregressive process).
e=0.2*randn(200,1);
u=randn(200,1);
stochastic_trend=cumsum(e);
deterministic_trend=.1*transpose(1:200);
x=zeros(200,1);
for i=2:200
    x(i)=.75*x(i-1)+u(i);
end
y=x+stochastic_trend+deterministic_trend;

%Instantiates time series objects.
ts0=dseries(y,'1950Q1');
ts1=dseries(x,'1950Q1');%stationarycomponent.

%Apply the Baxter-King filter.
ts2=ts0.baxter_king_filter();

%Plot the filtered time series.
plot(ts1(ts2.dates).data,'-');%Plot of the stationary component.
hold on
plot(ts2.data,'--r');            %Plot of the filtered y.
hold off
axis tight
id=get(gca,'XTick');
set(gca,'XTickLabel',strings(ts1.dates(id)));
```

Method:B=center(A[,geometric])

Method:center_(A[,geometric])

在 dseries 对象 A 中的变量围绕其算术平均值中心化处理，除非可选参数 geometric 设置为 true，在这种情况下，所有变量都除以它们的几何平均值。

Method:C=chain(A,B)

Method:chain_(A,B)

沿时间维度合并两个 dseries 对象。这两个对象必须具有相同数量的观测变量，并且 B 中的初始日期不得晚于 A 中的最后一个日期。返回的 dseries 对象 C 是通过使用 B 的累积增长因子扩展 A 来构建的。

示例

```
>>ts=dseries([1;2;3;4],dates('1950Q1'))

ts is a dseries object:

      |Variable_1
1950Q1|1
1950Q2|2
1950Q3|3
1950Q4|4

>>us=dseries([3;4;5;6],dates('1950Q3'))

us is a dseries object:

      |Variable_1
1950Q3|3
1950Q4|4
1951Q1|5
1951Q2|6

>>chain(ts,us)

ans is a dseries object:

      |Variable_1
```

```
1950Q1|1
1950Q2|2
1950Q3|3
1950Q4|4
1951Q1|5
1951Q2|6
```

Method:[error_flag,message]=check(A)

dseries 对象 A 的完整性检查。如果有错误，则返回 1，否则返回 0。第二个输出的变量是一个字符串，提供有关错误的简要信息。

Method:B=copy(A)

返回 A 的副本。如果对 A 应用了就地修改方法，则对象 B 不会受到影响。注意，如果将 A 分配给 C，C=A，则应用于 A 的任何就地修改方法都将改变 C。

示例

```
>>a=dseries(randn(5,1))

a is a dseries object:

  |Variable_1
1Y|-0.16936
2Y|-1.1451
3Y|-0.034331
4Y|-0.089042
5Y|-0.66997

>>b=copy(a);
>>c=a;
>>a.abs();
>>a.abs_();
>>a

a is a dseries object:
  |Variable_1
1Y|0.16936
2Y1.1451
```

```
3Y|0.034331
4Y|0.089042
5Y|0.66997

>>b

b is a dseries object:

  |Variable_1
1Y|-0.16936
2Y|-1.1451
3Y|-0.034331
4Y|-0.089042
5Y|-0.66997

>>c

c is a dseries object:

  |Variable_1
1Y|0.16936
2Y|1.1451
3Y|0.034331
4Y|0.089042
5Y|0.66997
```

Method:B=cumprod(A[,d[,v]])

Method:cumprod_(A[,d[,v]])

重载 dseries 对象的 MATLAB/Octave 的 cumprod 函数。如果 dseries 对象 A 中的变量具有 NaN，则无法计算累积乘积。如果将 dates 对象 d 提供为第二个参数，则该方法使得计算的累积乘积 dseries 对象 B 中的变量在日期 d 中标准化为 1。如果将单个观测值 dseries 对象 v 提供为第三个参数，则 B 中变量的累积乘积在日期 d 中被标准化为 v，即使得 B(d)匹配 v（dseries 对象 A 和 v 必须具有相同数量的变量）。

示例

```
>>ts1=dseries(2*ones(7,1));
```

```
>>ts2=ts1.cumprod();
>>ts2

ts2 is a dseries object:

  |cumprod(Variable_1)
1Y|2
2Y|4
3Y|8
4Y|16
5Y|32
6Y|64
7Y|128

>>ts3=ts1.cumprod(dates('3Y'));
>>ts3

ts3 is a dseries object:

  |cumprod(Variable_1)
1Y|0.25
2Y|0.5
3Y|1
4Y|2
5Y|4
6Y|8
7Y|16

>>ts4=ts1.cumprod(dates('3Y'),dseries(pi));
>>ts4

ts4 is a dseries object:

  |cumprod(Variable_1)
```

```
1Y|0.7854
2Y|1.5708
3Y|3.1416
4Y|6.2832
5Y|12.5664
6Y|25.1327
7Y|50.2655
```

Method:B=cumsum(A[,d[,v]])

Method:cumsum(A[,d[,v]])

重载 dseries 对象的 MATLAB/Octave 的 cumsum 函数。如果 dseries 对象 A 中的变量含有 NaN，则无法计算累积加和。如果将 dates 对象 d 提供为第二个参数，则该方法使得计算的累积加和 dseries 对象 B 中的变量在日期 d 中标准化为 0。如果将单个观测值 dseries 对象 v 提供为第三个参数，则 B 中的累积加和在日期 d 中被标准化为 v，即使得 B(d)匹配 v（dseries 对象 A 和 v 必须具有相同数量的变量）。

示例

```
>>ts1=dseries(ones(10,1));
>>ts2=ts1.cumsum();
>>ts2

ts2 is a dseries object:

   | cumsum(Variable_1)
1Y | 1
2Y | 2
3Y | 3
4Y | 4
5Y | 5
6Y | 6
7Y | 7
8Y | 8
9Y | 9
10Y| 10

>>ts3=ts1.cumsum(dates('3Y'));
```

```
>>ts3


ts3 is a dseries object:


   | cumsum(Variable_1)
1Y | -2
2Y | -1
3Y | 0
4Y | 1
5Y | 2
6Y | 3
7Y | 4
8Y | 5
9Y | 6
10Y| 7


>> ts4 = ts1.cumsum(dates('3Y'),dseries(pi));
>> ts4


ts4 is a dseries object:


   | cumsum(Variable_1)
1Y | 1.1416
2Y | 2.1416
3Y | 3.1416
4Y | 4.1416
5Y | 5.1416
6Y | 6.1416
7Y | 7.1416
8Y | 8.1416
9Y | 9.1416
10Y| 10.1416
```

Method:B=detrend(A,m)

Method:dentrend_(A,m)

使用 m 阶拟合多项式去除 dseries 对象 A 的趋势。注意，每个变量都使用不同的多项式去除趋势。

Method:B=dgrowth(A)

Method:dgrowth_(A)

计算日增长率。

Method:B=diff(A)

Method:diff_(A)

返回 dseries 对象 A 的一阶差分。

Method:disp(A)

重载 dseries 对象 MATLAB/Octave 的 disp 函数。

Method:display(A)

重载 dseries 对象的 MATLAB/Octave 的 display 函数。display 是 MATLAB 调用的函数，用于打印在 MATLAB 语句末尾缺少分号时对象的内容。如果 dseries 对象定义的时间跨度太大，则只会打印第一个和最后一个周期。如果 dseries 对象包含太多变量，则只会打印第一个和最后一个变量。如果需要所有期间和变量，则应改用 disp 方法。

Method:C=eq(A,B)

重载 MATLAB/Octave 的 eq（相等，==）运算符。dseries 对象 A 和 B 必须具有相同的观测时期（例如T）和变量个数（N）。返回的参数是一个$T \times N$的逻辑矩阵。当且仅当 A 和 B 中变量j的观测值i相同时，C 的元素(i,j)才等于 true。

示例

```
>>ts0=dseries(2*ones(3,1));
>>ts1=dseries([2;0;2]);
>>ts0==ts1


ans =


   3x1 logical array


   1
   0
   1
```

Method:l=exist(A,varname)

测试变量 varname 在 dseries 对象 A 中是否存在。当且仅当变量在 A 中存在时，返回 true。

示例

```
>>ts=dseries(randn(100,1));
>>ts.exist('Variable_1')

ans =

   logical

    1

>>ts.exist('Variable_2')

ans =

   logical

    0
```

Method:B=exp(A)

Method:exp_(A)

重载 dseries 对象的 MATLAB/Octave 的 exp 函数。

示例

```
>>ts0=dseries(rand(10,1));
>>ts1=ts0.exp();
```

Method:C=extract(A,B[,...])

从一个 dseries 对象 A 中提取一些变量并返回一个 dseries 对象 C。A 后面的输入参数是若干个字符串，表示要从 A 中提取到新的 dseries 对象 C 中的变量。为了简化子对象的创建，dseries 类重载了大括号（D=extract(A,B,C)等价于 D=A{B,C}）并允许隐式循环（在一对@符号之间定义，见下面的例子）或 MATLAB/Octave 的正则表达式（由方括号引入）。

示例

以下写法是等价的：

```
>>ts0=dseries(ones(100,10));
>>ts1=ts0{'Variable_1','Variable_2','Variable_3'};
>>ts2=ts0{'Variable_@1,2,3@'};
```

```
>>ts3=ts0{'Variable_[1-3]$'};
>>isequal(ts1,ts2)&&isequal(ts1,ts3)

ans =

   logical

    1
```

最多可以使用两个隐式循环来选择变量：

```
names={'GDP_1';'GDP_2';'GDP_3';'GDP_4';'GDP_5';'GDP_6';'GDP_7';'
GDP_8';'GDP_9';'GDP_10';'GDP_11';'GDP_12';...
'HICP_1';'HICP_2';'HICP_3';'HICP_4';'HICP_5';'HICP_6';'HICP_7';'
HICP_8';'HICP_9';'HICP_10';'HICP_11';'HICP_12'};

ts0=dseries(randn(4,24),dates('1973Q1'),names);
ts0{'@GDP,HICP@_@1,3,5@'}

ans is a dseries object:

      |GDP_1   |GDP_3   |GDP_5   |HICP_1 |HICP_3  |HICP_5
1973Q1|1.7906  |-1.6606 |-0.57716|0.60963|-0.52335|0.26172
1973Q2|2.1624  |3.0125  |0.52563 |0.70912|-1.7158 |1.7792
1973Q3|-0.81928|1.5008  |1.152   |0.2798 |0.88568 |1.8927
1973Q4|-0.03705|-0.35899|0.85838 |-1.4675|-2.1666 |-0.62032
```

Method:f=firstdate(A)

返回 dseries 对象 A 中的第一个时期。

Method:f=firstobservedperiod(A)

返回观测到 dseries 对象 A 中所有变量的第一个时期（非 NaN）。

Method:B=flip(A)

Method:flip_(A)

翻转数据成员的行（不改变时间顺序）

Method:f=frequency(B)

返回 dseries 对象 B 中变量的频率。

示例

```
>>ts=dseries(randn(3,2),'1973Q1');
>>ts.frequency

ans =

    4
```

Method:D=horzcat(A,B[,...])

重载 dseries 对象的 MATLAB/Octave 的 horzcat 方法。返回一个 dseries 对象 D，其包含在 dseries 对象中作为输入传递的变量：A、B……。如果输入未在相同的时间范围内定义，则该方法将 NaN 添加到变量中，以便在最小共同时间范围上重新定义变量。注意，作为输入传递的 dseries 对象中的名称必须不同，并且这些对象必须具有共同的频率。

示例

```
>>ts0=dseries(rand(5,2),'1950Q1',{'nifnif';'noufnouf'});
>>ts1=dseries(rand(7,1),'1950Q3',{'nafnaf'});
>>ts2=[ts0,ts1];
>>ts2

ts2 is a dseries object:

      |nifnif |noufnouf|nafnaf
1950Q1|0.17404|0.71431 |NaN
1950Q2|0.62741|0.90704 |NaN
1950Q3|0.84189|0.21854 |0.83666
1950Q4|0.51008|0.87096 |0.8593
1951Q1|0.16576|0.21184 |0.52338
1951Q2|NaN    |NaN     |0.47736
1951Q3|NaN    |NaN     |0.88988
1951Q4|NaN    |NaN     |0.065076
1952Q1|NaN    |NaN     |0.50946
```

Method:B=hpcycle(A[,lambda])

Method:hpcycle_(A[,lambda])

使用 Hodrick 和 Prescott（1997）滤波从 dseries 对象 A 中提取周期成分并返回 dseries 对象 B。平滑参数 lambda 的默认值为 1600。

示例

```
%Simulate a component model(stochastic trend,deterministic
%trend,and a stationary autoregressive process).
e=0.2*randn(200,1);
u=randn(200,1);
stochastic_trend=cumsum(e);
deterministic_trend=.1*transpose(1:200);
x=zeros(200,1);
for i=2:200
    x(i)=.75*x(i-1)+u(i);
end
y=x+stochastic_trend+deterministic_trend;

%Instantiates time series objects.
ts0=dseries(y,'1950Q1');
ts1=dseries(x,'1950Q1');%stationary component.

%Apply the HP filter.
ts2=ts0.hpcycle();

%Plot the filtered time series.
plot(ts1(ts2.dates).data,'-k');%Plot of the stationary component.
hold on
plot(ts2.data,'--r');%Plot of the filtered y.
hold off
axis tight
id=get(gca,'XTick');
set(gca,'XTickLabel',strings(ts.dates(id)));
```

Method:B=hptrend(A[,lambda])

Method:hptrend_(A[,lambda])

使用 Hodrick 和 Prescott（1997）滤波从 dseries 对象 A 中提取趋势成分，并返回 dseries 对象 B。平滑参数 lambda 的默认值为 1600。

示例

```
%Using the same generating data process
```

```
%as in the previous example:

ts1=dseries(stochastic_trend+deterministic_trend,'1950Q1');
%Apply the HP filter.
ts2=ts0.hptrend();

%Plot the filtered time series.
plot(ts1.data,'-k');%Plot of the nonstationary components.
hold on
plot(ts2.data,'--r');%Plot of the estimated trend.
hold off
axis tight
id=get(gca,'XTick');
set(gca,'XTickLabel',strings(ts0.dates(id)));
```

Method:C=insert(A,B,I)

将包含在 dseries 对象 B 中的变量插入到 dseries 对象 A 中由向量 I 的整数标量指定的位置，返回增广的 dseries 对象 C。I 中的整数标量必须取值介于``和 A.length()+1 之间，且其指代的是 A 的列数。dseries 对象 A 和 B 不需要定义在相同的时间范围内，但它们需要具有共同的频率。

示例

```
>>ts0=dseries(ones(2,4),'1950Q1',{'Sly';'Gobbo';'Sneaky';'Stealthy'});
>>ts1=dseries(pi*ones(2,1),'1950Q1',{'Noddy'});
>>ts2=ts0.insert(ts1,3)

ts2 is a dseries object:

      |Sly|Gobbo|Noddy |Sneaky|Stealthy
1950Q1|1  |1    |3.1416|1     |1
1950Q2|1  |1    |3.1416|1     |1

>>ts3=dseries([pi*ones(2,1) sqrt(pi)*ones(2,1)],'1950Q1',{'Noddy';'Tessie Bear'});
>>ts4=ts0.insert(ts1,[3,4])
```

```
ts4 is a dseries object:

      |Sly|Gobbo|Noddy |Sneaky|Tessie Bear|Stealthy
1950Q1|1  |1    |3.1416|1     |1.7725     |1
1950Q2|1  |1    |3.1416|1     |1.7725     |1
```

Method:B=isempty(A)

重载 MATLAB/Octave 的 isempty 函数。如果 dseries 对象 A 为空，则返回 true。

Method:C=isequal(A,B)

重载 MATLAB/Octave 的 isequal 函数。如果 dseries 对象 A 和 B 相同，则返回 true。

Method:C=isinf(A)

重载 MATLAB/Octave 的 isinf 函数。返回一个逻辑数组，其中当且仅当变量 j 在周期 A.dates(i)中是有限的时，元素(i,j)等于 true。

Method:C=isnan(A)

重载 MATLAB/Octave 的 isnan 函数。返回一个逻辑数组，其中当且仅当变量 j 在周期 A.dates(i)中不是 NaN 时，元素(i,j)等于 true。

Method:C=isreal(A)

重载 MATLAB/Octave 的 isreal 函数。返回一个逻辑数组，其中当且仅当变量 j 在周期 A.dates(i)中为实数时，元素(i,j)等于 true。

Method:B=lag(A[,p])

Method:lag_(A[,p])

返回滞后时间序列。整数标量 p，即滞后阶数的默认值为 1。

示例

```
>>ts0=dseries(transpose(1:4),'1950Q1')

ts0 is a dseries object:

      |Variable_1
1950Q1|1
1950Q2|2
1950Q3|3
1950Q4|4
```

```
>>ts1=ts0.lag()

ts1 is a dseries object:

      |Variable_1
1950Q1|NaN
1950Q2|1
1950Q3|2
1950Q4|3
>>ts2=ts0.lag(2)

ts2 is a dseries object:

      |Variable_1
1950Q1|NaN
1950Q2|NaN
1950Q3|1
1950Q4|2

%dseries class overloads the parenthesis
%so that ts.lag(p) can be written more
%compactly as ts(-p). For instance:

>>ts0.lag(1)

ans is a dseries object:

      |Variable_1
1950Q1|NaN
1950Q2|1
1950Q3|2
1950Q4|3
```

或者另一种写法：

```
>>ts0(-1)
```

```
ans is a dseries object:

      |Variable_1
1950Q1|NaN
1950Q2|1
1950Q3|2
1950Q4|3
```

Method:l=lastdate(B)

返回 dseries 对象 B 中的最后一个时期。

示例

```
>>ts=dseries(randn(3,2),'1973Q1');
>>ts.lastdate()

ans=<dates:1973Q3>
```

Method:f=lastobservedperiod(A)

返回到 dseries 对象 A 中所有变量都被观测到（非 NaN）的最后一个时期。

Method:B=lead(A[,p])

Method:lead_(A[,p])

返回超前时间序列。整数标量 p，即超前阶数的默认值是 1。在 lag 方法中，dseries 类重载了小括号，以便 ts.lead(p)等价于 ts(p)。

示例

```
>>ts0=dseries(transpose(1:4),'1950Q1');
>>ts1=ts0.lead()

ts1 is a dseries object:

      |Variable_1
1950Q1|2
1950Q2|3
1950Q3|4
1950Q4|NaN

>>ts2=ts0(2)
```

```
ts2 is a dseries object:

      |Variable_1
1950Q1|3
1950Q2|4
1950Q3|NaN
1950Q4|NaN
```

备注

dseries 对象括号的重载允许通过复制/粘贴 model 模块中声明的方程轻松创建新的 dseries 对象。例如，如果在 model 模块中定义了一个欧拉方程：

```
model;
...
1/C-beta/C(1)*(exp(A(1))*K^(alpha-1)+1-delta);
...
end;
```

如果变量，``A 和 K 被定义为 dseries 对象，则通过编写：

```
Residuals=1/C-beta/C(1)*(exp(A(1))*K^(alpha-1)+1-delta);
```

在 model 模块之外，我们为欧拉方程的残差创建了一个新的 dseries 对象，称之为 Residuals（模型模块中定义的方程的条件期望为零，但残差不为零）。

Method:B=lineartrend(A)

返回以 0 为中心的线性趋势，趋势的长度由 dseries 对象 A 的大小（周期数）给出。

示例

```
>>ts=dseries(ones(3,1));
>>ts.lineartrend()

ans =
     -1
      0
      1
```

Method:B=log(A)

Method:log_(A)

重载 dseries 对象的 MATLAB/Octave 的 log 函数。

示例

```
>>ts0=dseries(rand(10,1));
>>ts1=ts0.log();
```

Method:B=mdiff(A)

Method:mdiff_(A)

Method:B=mgrowth(A)

Method:mgrowth_(A)

计算 dseries 对象 A 中变量的月度差分或月度增长率。

Method:B=mean(A[,geometric])

重载 dseries 对象的 MATLAB/Octave 的 mean 函数。返回 dseries 对象 A 中每个变量的平均值。如果第二个参数为 true，则计算几何平均值，否则默认报告算术平均值。

Method:C=merge(A,B[,legacy])

在 dseries 对象 C 中合并两个 dseries 对象 A 和 B。对象 A 和 B 需要具有共同的频率，但可以在不同的时间范围内定义。如果在 dseries 对象 A 和 B 中都定义了同一个变量 x，则 merge 将选择在第二个输入参数 B 中定义的变量 x，除非 B 中存在取值为 NaN 的元素，而 A 中的相应元素（即相同期间）是明确定义的数字。可以通过将可选参数 legacy 设置为 true 来更改此行为，在这种情况下，即使第二个变量具有 NaN，第二个变量也会覆盖第一个变量。

示例

```
>>ts0=dseries(rand(3,2),'1950Q1',{'A1';'A2'})


ts0 is a dseries object:


      |A1     |A2
1950Q1|0.96284|0.5363
1950Q2|0.25145|0.31866
1950Q3|0.34447|0.4355


>>ts1=dseries(rand(3,1),'1950Q2',{'A1'})


ts1 is a dseries object:


      |A1
1950Q2|0.40161
1950Q3|0.81763
```

```
1950Q4|0.97769

>>merge(ts0,ts1)

ans is a dseries object:

      |A1     |A2
1950Q1|0.96284|0.5363
1950Q2|0.40161|0.31866
1950Q3|0.81763|0.4355
1950Q4|0.97769|NaN

>>merge(ts1,ts0)

ans is a dseries object:

      |A1     |A2
1950Q1|0.96284|0.5363
1950Q2|0.25145|0.31866
1950Q3|0.34447|0.4355
1950Q4|0.97769|NaN
```

Method:C=minus(A,B)

用于 dseries 对象的重载 MATLAB/Octave 的 minus（-）运算符，逐个元素相减。如果 A 和 B 都是 dseries 对象，则不需要在相同的时间范围内定义。如果 A 和 B 是具有T_A和T_B个观测值以及N_A和N_B个变量的 dseries 对象，则N_A必须等于N_B或1，并且N_B必须等于N_A或1。如果$T_A = T_B$，isequal(A.init,B.init)返回 1，并且$N_A = N_B$，则接着 minus 运算符将计算每对(t, n)，其中$1 \le t \le T_A$且$1 \le n \le N_A$，C.data(t,n)=A.data(t,n)-B.data(t,n)。如果$N_B = 1$且$N_A > 1$，则较小的 dseries 对象（B）在较大的 dseries 对象（A）上“传播”，以便它们具有兼容的形状，minus 运算符将从 A 中每个变量减去 B 中定义的变量。如果 B 是双标量，那么 minus 法将从 A 中的所有观测值/变量中减去 B。如果 B 是长度为N_A的行向量，那么对于$i = 1, \ldots, N_A$，minus 法将从变量 i 中的每一个观测值减去 B(i)。如果 B 是长度为T_A的列向量，则 minus 法将在每一个变量中减去 B。

示例

```
>>ts0=dseries(rand(3,2));
>>ts1=ts0{'Variable_2'};
>>ts0-ts1

ans is a dseries object:

  |Variable_1|Variable_2
1Y|-0.48853  |0
2Y|-0.50535  |0
3Y|-0.32063  |0

>>ts1

ts1 is a dseries object:

  |Variable_2
1Y|0.703
2Y|0.75415
3Y|0.54729

>>ts1-ts1.data(1)

ans is a dseries object:

  |Variable_2
1Y|0
2Y|0.051148
3Y|-0.15572

>>ts1.data(1)-ts1

ans is a dseries object:

  |Variable_2
```

```
1Y|0
2Y|-0.051148
3Y|0.15572
```

Method:C=mpower(A,B)

重载 dseries 对象的 MATLAB/Octave 的 mpower(^)运算符并计算逐个元素的幂。A 是具有 N 个变量和 T 个观测值的 dseries 对象。如果 B 是实数标量，则 mpower(A,B)返回一个 dseries 对象 C，其中 C.data(t,n)=A.data(t,n)^C。如果 B 是具有 N 个变量和 T 个观测值的 dseries 对象，则 mpower(A,B)返回一个 dseries 对象 C，其中 C.data(t,n)=A.data(t,n)^C.data(t,n)。

示例

```
>>ts0=dseries(transpose(1:3));
>>ts1=ts0^2

ts1 is a dseries object:

  |Variable_1
1Y|1
2Y|4
3Y|9

>>ts2=ts0^ts0

ts2 is a dseries object:

  |Variable_1
1Y|1
2Y|4
3Y|27
```

Method:C=mrdivide(A,B)

用于 dseries 对象的重载 MATLAB/Octave 的 mrdivide（/）运算符，逐个元素相除（类似于 MATLAB/Octave 运算符./）。如果 A 和 B 都是 dseries 对象，则不需要在相同的时间范围内定义。如果 A 和 B 是具有T_A和T_B个观测值以及N_A和N_B个变量的 dseries 对象，则N_A必须等于N_B或1，并且N_B必须等于N_A或1。如果$T_A = T_B$，isequal(A.init, B.init)返回 1，并且$N_A = N_B$，则接着 mrdivide 运算符将计算每对(t, n)，其中$1 \leq t \leq$

T_A且$1 \le n \le N_A$，C.data(t,n)=A.data(t,n)/B.data(t,n)。如果$N_B = 1$且$N_A > 1$，则较小的 dseries(B)在较大的 dseries(A)上“传播”，以便它们具有兼容的形状。在这种情况 mrdivide 运算符将从 A 中每个变量除以 B 中定义的变量。如果 B 是双标量，那么 mrdivide 将从 A 中的所有观测值/变量中除以 B。如果 B 是长度为N_A的行向量，那么对于$i = 1, \dots, N_A$，mrdivide 将从变量 i 中的每一个观测值都除以 B(i)。如果 B 是长度为T_A的列向量，则 mrdivide 法将在每一个变量中逐个元素除以 B。

示例

```
>>ts0=dseries(rand(3,2))

ts0 is a dseries object:

  |Variable_1|Variable_2
1Y|0.72918   |0.90307
2Y|0.93756   |0.21819
3Y|0.51725   |0.87322

>>ts1=ts0{'Variable_2'};
>>ts0/ts1

ans is a dseries object:

  |Variable_1|Variable_2
1Y|0.80745   |1
2Y|4.2969    |1
3Y|0.59235   |1
```

Method:C=mtimes(A,B)

用于 dseries 对象的重载 MATLAB/Octave 的 mtimes（*）运算符和 Hadammard 乘积（类似于 MATLAB/Octave 运算符.*）。如果 A 和 B 都是 dseries 对象，则不需要在相同的时间范围内定义。如果 A 和 B 是具有T_A和T_B个观测值以及N_A和N_B个变量的 dseries 对象，则N_A必须等于N_B或1，并且N_B必须等于N_A或1。如果$T_A = T_B$，isequal(A.init, B.init)返回1，并且$N_A = N_B$，则接着 mtimes 运算符将对每对(t,n)进行计算，其中$1 \le t \le T_A$且$1 \le n \le N_A$，C.data(t,n)=A.data(t,n)*B.data(t,n)。如果$N_B = 1$且$N_A > 1$，则较小的 dseries(B)在较大的 dseries(A)上“传播”，以便它们具有兼容的形状。在这种情况 mtimes 运算符将从 A 中每个变量乘以 B 中定义的变量。如果 B 是双

标量，那么 mtimes 将从 A 中的所有观测值/变量中乘以 B。如果 B 是长度为N_A的行向量，那么对于$i = 1, \ldots, N_A$，mtimes 将从变量 i 中的每一个观测值都乘以 B(i)。如果 B 是长度为T_A的列向量，则 mtimes 法将在每一个变量中逐个元素乘以 B。

Method:B=nanmean(A[,geometric])

用于 dseries 对象的重载 MATLAB/Octave 的 nanmean 函数。返回 dseries 对象 A 中每个变量的平均值，忽略 NaN 值。如果第二个参数为 true，则计算几何平均值，否则默认报告算术平均值。

Method:B=nanstd(A[,geometric])

用于 dseries 对象的重载 MATLAB/Octave 的 nanstd 函数。返回 dseries 对象 A 中每个变量的标准差，忽略 NaN 值。如果第二个参数为 true，则计算几何标准差，第二的参数默认值是 false。

Method:C=ne(A,B)

重载 MATLAB/Octave 的 ne（不等于，~=）运算符。dseries 对象 A 和 B 必须具有相同数量的观测值（即T）和变量（N）。返回的参数是一个由 0 和 1 组成的$T \times N$矩阵。当且仅当 A 和 B 中变量j的观测值i不相等时，C 的元素(i, j)等于 1。

示例

```
>>ts0=dseries(2*ones(3,1));
>>ts1=dseries([2;0;2]);
>>ts0~=ts1


ans=


  3x1 logical array
   0
   1
   0
```

Method:B=nobs(A)

返回 dseries 对象 A 中的观测数。

示例

```
>>ts0=dseries(randn(10));
>>ts0.nobs


ans =
```

```
    10
```

Method:B=onesidedhpcycle(A[,lambda[,init]])

Method:onesidedhpcycle_(A[,lambda[,init]])

使用单边 HP 滤波（带有卡尔曼滤波）从 dseries 对象 A 中提取周期成分并返回 dseries 对象 B。平滑参数 lambda 的默认值为 1600。默认情况下，如果未提供 init，初始值基于前两个观测值。

Method:B=onesidedhptrend(A[,lambda[,init]])

Method:onesidedhptrend_(A[,lambda[,init]])

使用单边 HP 滤波（带有卡尔曼滤波）从 dseries 对象 A 中提取趋势成分并返回 dseries 对象 B。平滑参数 lambda 的默认值为 1600。默认情况下，如果未提供 init，初始值基于前两个观测值。

Method:h=plot(A)

Method:h=plot(A,B)

Method:h=plot(A[,...])

Method:h=plot(A,B[,...])

用于 dseries 对象的重载 MATLAB/Octave 的 plot 函数。返回一个 MATLAB/Octave 绘图句柄，可用于修改绘制的时间序列属性。如果只有一个 dseries 对象 A 作为参数传递，则绘图函数会将关联的日期放在 x 横坐标上。如果这个 dseries 对象只包含一个变量，则可以传递额外的参数来修改绘图属性（就像使用 MATLAB/Octave 版本的绘图函数所做的那样）。如果 dseries 对象 A 包含多个变量，则无法传递这些附加参数，并且必须使用返回的绘图句柄和 MATLAB/Octave 的 set 函数修改绘制的时间序列属性（参见下面的示例）。如果两个 dseries 对象 A 和 B 作为输入参数传递，绘图函数将根据 B 中的变量绘制 A 中的变量（每个对象中的变量数必须相同，否则会报错）。同样，如果每个对象只包含一个变量，则可以传递额外的参数来修改绘制的时间序列的属性，否则必须使用 MATLAB/Octave 的 set 命令。

示例

定义一个带有两个变量（默认命名为 Variable_1 和 Variable_2）的 dseries 对象：

```
>>ts=dseries(randn(100,2),'1950Q1');
```

以下命令将绘制 ts 中的第一个变量：

```
>>plot(ts{'Variable_1'},'-k','linewidth',2);
```

下一个命令将在同一图上绘制 ts 中的所有变量：

```
>>h=plot(ts);
```

如果想要修改绘制的时间序列的属性（线型、颜色等），可以使用 set 函数（参见 M

ATLAB 的文档）：

```
>>set(h(1),'-k','linewidth',2);
>>set(h(2),'--r');
```

以下命令将针对 exp(Variable_1)绘制 Variable_1：

```
>>plot(ts{'Variable_1'},ts{'Variable_1'}.exp(),'ok');
```

同样，也可以使用返回的绘图句柄和 set 函数修改属性：

```
>>h=plot(ts,ts.exp());
>>set(h(1),'ok');
>>set(h(2),'+r');
```

Method:C=plus(A,B)

用于 dseries 对象的重载 MATLAB/Octave 的 plus（+）运算符，逐个元素相加。如果 A 和 B 都是 dseries 对象，则不需要在相同的时间范围内定义。如果 A 和 B 是具有T_A和T_B个观测值以及N_A和N_B个变量的 dseries 对象，则N_A必须等于N_B或1，并且N_B必须等于N_A或1。如果$T_A = T_B$，isequal(A.init,B.init)返回1，并且$N_A = N_B$，则接着 plus 运算符将对每对(t,n)进行计算，其中$1 \le t \le T_A$且$1 \le n \le N_A$，C.data(t,n)=A.data(t,n)+B.data(t,n)。如果$N_B = 1$且$N_A > 1$，则较小的 dseries(B)在较大的 dseries(A)上“传播”，以便它们具有兼容的形状，plus 运算符将从 A 中每个变量加上 B 中定义的变量。如果 B 是双标量，那么 plus 将从 A 中的所有观测值/变量中加上 B。如果 B 是长度为N_A的行向量，那么对于$i = 1, \ldots, N_A$，plus 将从变量 i 中的每一个观测值都加上 B(i)。如果 B 是长度为T_A的列向量，则 plus 法讲在每一个变量中加上 B。

Method:C=pop(A[,B])

Method:pop_(A[,B])

从 dseries 对象 A 中删除变量 B。默认情况下，如果未提供第二个参数，则删除最后一个变量。

示例

```
>>ts0=dseries(ones(3,3));
>>ts1=ts0.pop('Variable_2');


ts1 is a dseries object:


  |Variable_1|Variable_3
1Y|1         |1
2Y|1         |1
3Y|1         |1
```

Method:A=projection(A,info,periods)

在 dseries 对象 A.info 的预测变量是一个$n \times 3$元胞数组。每行提供了预测变量的必要信息。第一列包含了变量的名称（行、字符、数组），第二列包含了预测关联变量（行、字符、数组）的方法名称，可行值包括“Trend”、“Constant”和“AR”。最后一列提供了预测的数量信息。如果第二列是“Trend”，第三列就是（指数）趋势增长因子；第二列是“Constant”，第三列是变量的水平值；第二列是“AR”，第三列就是自回归参数。如果第三列包含一个$1 \times p$双精度向量，变量可以用 AR(p)模型预测。不检验 AR(p)模型的平稳性。常数预测的情况下，使用变量的最后一个值，包含“Trend”和一个等于 1 的增长因子，或者包含等于 1（随机游走）的自回归参数的“AR”。该预测程序只处理指数趋势。

示例

```
>>data=ones(10,4);
>>ts=dseries(data,'1990Q1',{'A1','A2','A3','A4'});
>>info={'A1','Trend',1.2;'A2','Constant',0.0;'A3','AR',5;'A4','AR
',[.4,-.2]};
>>ts.projection(info,10);
```

Method:B=qdiff(A)

Method:B=qgrowth(A)

Method:qdiff_(A)

Method:qgrowth_(A)

计算季度差分或增长率。

示例

```
>>ts0=dseries(transpose(1:4),'1950Q1');
>>ts1=ts0.qdiff()


ts1 is a dseries object:


      |Variable_1
1950Q1|NaN
1950Q2|1
1950Q3|1
1950Q4|1


>>ts0=dseries(transpose(1:6),'1950M1');
>>ts1=ts0.qdiff()
```

```
ts1 is a dseries object:

      |Variable_1
1950M1|NaN
1950M2|NaN
1950M3|NaN
1950M4|3
1950M5|3
1950M6|3
```

Method:C=remove(A,B)

Method:remove_(A,B)

带有两个参数的 pop 方法的别名。从 dseries 对象 A 中删除变量 B。

示例

```
>>ts0=dseries(ones(3,3));
>>ts1=ts0.remove('Variable_2');

ts1 is a dseries object:

  |Variable_1|Variable_3
1Y|1         |1
2Y|1         |1
3Y|1         |1
```

可以使用更短的语法：remove(ts,'Variable_2')等价于 ts{'Variable_2'}=[]（[]可以被任何空对象替换）。如果必须删除多个变量，则此替代语法很有用。例如：

```
ts{'Variable_@2,3,4@'}=[];
```

将从 dseries 对象 ts 中删除 Variable_2、Variable_3 和 Variable_4（如果这些变量存在）。不能使用正则表达式，但可以使用隐式循环。

Method:B=rename(A,oldname,newname)

Method:rename_(A,oldname,newname)

在 dseries 对象 A，将变量 oldname 重命名为 newname。返回一个 dseries 对象。如果需要重命名多个变量，则可以将字符数组的单元格作为第二个和第三个参数传递。

示例

```
>>ts0=dseries(ones(2,2));
```

```
>>ts1=ts0.rename('Variable_1','Stinkly')


ts1 is a dseries object:


  |Stinkly|Variable_2
1Y|1      |1
2Y|1      |1
```

Method:C=rename(A,newname)

Method:rename_(A,newname)

将 A 中的名称替换为在元胞字符串数组 newname 中传递的名称。newname 的元素数目和 dseries 对象 A 变量数相同。返回一个 dseries 对象。

示例

```
>>ts0=dseries(ones(2,3));
>>ts1=ts0.rename({'TinkyWinky','Dipsy','LaaLaa'})


ts1 is a dseries object:


  |TinkyWinky|Dipsy|LaaLaa
1Y|1         |1    |1
2Y|1         |1    |1
```

Method: A=resetops(A,ops)

重新定义 ops 成员。

Method: A=resetags(A,ops)

重新定义 tags 成员。

Method:B=round(A[,n])

Method:round_(A[,n])

近似为最近的小数或整数。n 是精度参数（小数位数）。默认值为 0，表示方法默认为近似为最近的整数。

示例

```
>>ts=dseries(pi)


ts is a dseries object:


   | Variable_1
```

```
1Y | 3.1416


>>ts.round_();
>>ts


ts is a dseries object:


   | Variable_1
1Y | 3
```

Method:save(A,basename[,format])

重载 MATLAB/Octave 的 save 函数并将 dseries 对象 A 保存到磁盘。可行的格式有 mat（这是默认值）、m（MATLAB/Octave 脚本）和 csv（MATLAB 二进制数据文件）。没有扩展名的文件名由 basename 指定。

示例

```
>>ts0=dseries(ones(2,2));
>>ts0.save('ts0','csv');
```

最后一个命令将创建一个文件 ts0.csv，内容如下：

```
,Variable_1,Variable_2
1Y,               1,               1
2Y,               1,               1
```

要创建 MATLAB/Octave 脚本，使用以下命令：

```
>>ts0.save('ts0','m');
```

将生成一个包含以下内容的文件 ts0.m：

```
%File created on 14-Nov-2013 12:08:52.


FREQ__=1;
INIT__='1Y';


NAMES__={'Variable_1';'Variable_2'};
TEX__={'Variable_{1}';'Variable_{2}'};
OPS__={};
TAGS__=struct();


Variable_1=[
```

```
            1
            1];


Variable_2=[
            1
            1];
```

如上所述，在实例化 dseries 对象时可以加载生成的（csv、m 或 mat）文件。

Method:B=set_names(A,s1,s2,...)

重命名 dseries 对象 A 中的变量并返回具有新名称 s1、s2……的 dseries 对象 B。第一个（dseries 对象 A）之后的输入参数的数量必须等于 A.vobs（A 中的变量数量）。s1 将是 B 中第一个变量的名称，s2 将是 B 中第二个变量的名称，依此类推。

示例

```
>>ts0=dseries(ones(1,3));
>>ts1=ts0.set_names('Barbibul',[],'Barbouille')


ts1 is a dseries object:


   |Barbibul|Variable_2|Barbouille
1Y |1       |1         |1
```

Method:[T,N]=size(A[,dim])

重载 MATLAB/Octave 的 size 函数。返回 dseries 对象 A 中的观测数（即 A.nobs）和变量数（即 A.vobs）。如果传递了第二个输入参数，则 size 函数在 dim=1 时返回观测的数量或在 dim=2 时返回变量的数量（对于所有其他的 dim 值，会发出错误）。

示例

```
>>ts0=dseries(ones(1,3));
>>ts0.size()


ans =


      1     3
```

Method:B=std(A[,geometric])

重载 dseries 对象的 MATLAB/Octave 的 std 函数。返回 dseries 对象 A 中每个变量的标准差。如果第二个参数为 true，则计算几何标准差（第二个参数的默认值为 false）。

Method:B=subsample(A,d1,d2)

返回一个时期在 dates 定义的介于 d1 到 d2 的子样本。同样可以通过使用 date 对象为 dseries 对象建立索引来实现，但是 subsample 方法更容易通过编程的方式实现。

示例

```
>>o=dseries(transpose(1:5));
>>o.subsample(dates('2y'),dates('4y'))


ans is a dseries object:


  |Variable_1
2Y|2
3Y|3
4Y|4
```

Method:A=tag(A,a[,b,c])

将标签添加到 dseries 对象 A 中的变量。

示例

```
>>ts=dseries(randn(10,3));
>>tag(ts,'type');            %Define a tag name.
>>tag(ts,'type','Variable_1','Stock');
>>tag(ts,'type','Variable_2','Flow');
>>tag(ts,'type','Variable_3','Stock');
```

Method:B=tex_rename(A,name,newtexname)

Method:B=tex_rename(A,newtexname)

Method:tex_rename_(A,name,newtexname)

Method:tex_rename_(A,newtexname)

将变量 name 的 tex 名称重新定义为 dseries 对象 A 中的 newtexname。返回一个 dseries 对象。

只有两个参数 A 和 newtexname，A 的 tex 名称重新定义为包含在 newtexname 中的名称。此处的 newtexname 是一个元胞字符串数组，其条目数与 A 中的变量数相同。

Method:B=uminus(A)

用于 dseries 对象的重载 uminus（-，一元减法运算符）。

示例

```
>>ts0=dseries(1)
```

```
ts0 is a dseries object:

  |Variable_1
1Y|1

>>ts1=-ts0

ts1 is a dseries object:

  |Variable_1
1Y|-1
```

Method:D=vertcat(A,B[,...])

用于 dseries 对象的重载 MATLAB/Octave 的 vertcat 函数。此方法用于将更多的观测值附加到一个 dseries 对象。返回一个 dseries 对象 D，其中包含作为输入传递的 dseries 对象中的变量。所有的输入参数必须是 dseries 对象，其包含相同的变量且定义在不同的时期范围内。

示例

```
>>ts0=dseries(rand(2,2),'1950Q1',{'nifnif';'noufnouf'});
>>ts1=dseries(rand(2,2),'1950Q3',{'nifnif';'noufnouf'});
>>ts2=[ts0;ts1]

ts2 is a dseries object:

      |nifnif   |noufnouf
1950Q1|0.82558 |0.31852
1950Q2|0.78996 |0.53406
1950Q3|0.089951|0.13629
1950Q4|0.11171 |0.67865
```

Method:B=vobs(A)

返回 dseries 对象 A 中变量的数量。

示例

```
>>ts0=dseries(randn(10,2));
>>ts0.vobs
```

```
ans =


    2
```

Method:B=ydiff(A)

Method:B=ygrowth(A)

Method:ydiff_(A)

Method:ygrowth_(A)

计算年度差分或增长率。

6.3 X-13 ARIMA-SEATS 接口

Dynare class:x13

x13 类为 X-13 ARIMA-SEATS 参考手册中记载的每个 X-13 命令提供一个方法（*x11*、*automdl*、*estimate*……），然后选项可以通过 key/value 对传递。x13 类有 22 个成员：

成员

- **y**：带有单变量的 dseries 对象；
- **x**：带有任意数量变量的 dseries 对象（用于 REGRESSION 模块）；
- **arima**：包含 ARIMA 模型命令选项的结构；
- **automdl**：包含 ARIMA 模型选择命令选项的结构；
- **regression**：包含 Regression 命令选项的结构；
- **estimate**：包含 estimation 命令选项的结构；
- **transform**：包含 transform 命令选项的结构；
- **outlier**：包含 outlier 命令选项的结构；
- **forecast**：包含 forecast 命令选项的结构；
- **check**：包含 check 命令选项的结构；
- **x11**：包含 X11 命令选项的结构；
- **force**：包含 force 命令选项的结构；
- **history**：包含 history 命令选项的结构；
- **metadata**：包含 metadata 命令选项的结构；
- **identify**：包含 identify 命令选项的结构；
- **pickmdl**：包含 pickmdl 命令选项的结构；
- **seats**：包含 seats 命令选项的结构；
- **slidingspans**：包含 slidingspans 命令选项的结构；
- **spectrum**：包含 spectrum 命令选项的结构；
- **x11regression**：包含 x11Regression 命令选项的结构；
- **results**：包含 x13 返回结果的结构；

- **commands：**包含一组命令的元胞数组。

所有这些成员都是用户自定义的。以下是可用的构造函数：

Constructor:x13(y)

用 dseries 对象 y 实例化一个 x13 对象。传递为参数的 dseries 对象必须只包含一个变量，其需要传递给 X-13。

Constructor:x13(y,x)

用 dseries 对象 y 和 x 实例化一个 x13 对象。作为参数传递的第一个 dseries 对象必须只包含一个变量，第二个 dseries 对象包含 X-13 命令使用的一些外生变量。两个对象必须定义在相同的时期上。

以下方法可以设置 X-13 命令的序列，编写一个*.spc 文件并运行 X-13 二进制文件：

Method:A=arima(A,key,value[,key,value[,[...]]])

接口到 arima 命令，参见 X-13 ARIMA-SEATS 手册。所有的选项必须通过 key/value 对来传递。

Method:A=automdl(A,key,value[,key,value[,[...]]])

接口到 automdl 命令，参见 X-13 ARIMA-SEATS 手册。所有的选项必须通过 key/value 对来传递。

Method:A=regression(A,key,value[,key,value[,[...]]])

接口到 regression 命令，参见 X-13 ARIMA-SEATS 手册。所有的选项必须通过 key/value 对来传递。

Method:A=estimate(A,key,value[,key,value[,[...]]])

接口到 estimate 命令，参见 X-13 ARIMA-SEATS 手册。所有的选项必须通过 key/value 对来传递。

Method:A=transform(A,key,value[,key,value[,[...]]])

接口到 transform 命令，参见 X-13 ARIMA-SEATS 手册。所有的选项必须通过 key/value 对来传递。

Method:A=outlier(A,key,value[,key,value[,[...]]])

接口到 outlier 命令，参见 X-13 ARIMA-SEATS 手册。所有的选项必须通过 key/value 对来传递。

Method:A=forecast(A,key,value[,key,value[,[...]]])

接口到 forecast 命令，参见 X-13 ARIMA-SEATS 手册。所有的选项必须通过 key/value 对来传递。

Method:A=check(A,key,value[,key,value[,[...]]])

接口到 check 命令，参见 X-13 ARIMA-SEATS 手册。所有的选项必须通过 key/value 对来传递。

Method:A=x11(A,key,value[,key,value[,[...]]])

接口到 x11 命令，参见 X-13 ARIMA-SEATS 手册。所有的选项必须通过 key/value 对来传递。

Method:A=force(A,key,value[,key,value[,[...]]])

接口到 force 命令，参见 X-13 ARIMA-SEATS 手册。所有的选项必须通过 key/value 对来传递。

Method:A=history(A,key,value[,key,value[,[...]]])

接口到 history 命令，参见 X-13 ARIMA-SEATS 手册。所有的选项必须通过 key/value 对来传递。

Method:A=metadata(A,key,value[,key,value[,[...]]])

接口到 metadata 命令，参见 X-13 ARIMA-SEATS 手册。所有的选项必须通过 key/value 对来传递。

Method:A=identify(A,key,value[,key,value[,[...]]])

接口到 identify 命令，参见 X-13 ARIMA-SEATS 手册。所有的选项必须通过 key/value 对来传递。

Method:A=pickmdl(A,key,value[,key,value[,[...]]])

接口到 pickmdl 命令，参见 X-13 ARIMA-SEATS 手册。所有的选项必须通过 key/value 对来传递。

Method:A=seats(A,key,value[,key,value[,[...]]])

接口到 seats 命令，参见 X-13 ARIMA-SEATS 手册。所有的选项必须通过 key/value 对来传递。

Method:A=slidingspans(A,key,value[,key,value[,[...]]])

接口到 slidingspans 命令，参见 X-13 ARIMA-SEATS 手册。所有的选项必须通过 key/value 对来传递。

Method:A=spectrum(A,key,value[,key,value[,[...]]])

接口到 spectrum 命令，参见 X-13 ARIMA-SEATS 手册。所有的选项必须通过 key/value 对来传递。

Method:A=x11regression(A,key,value[,key,value[,[...]]])

接口到 x11regression 命令，参见 X-13 ARIMA-SEATS 手册。所有的选项必须通过 key/value 对来传递。

Method:print(A[,basefilename])

打印一个包含所有 X-13 命令的*.spc 文件。可选的第二个参数是一个指定文件名称（不包括扩展名）的行字符数组，。

Method:run(A)

调用 X-13 二进制文件并运行先前定义的命令。所有的结果都储存在 A.results 结构中。当有意义时，这些结果被保存在 dseries 对象中（例如预测或滤波变量）。

示例

```
>>ts=dseries(rand(100,1),'1999M1');
>>o=x13(ts);


>>o.x11('save','(d11)');
>>o.automdl('savelog','amd','mixed','no');
>>o.outlier('types','all','save','(fts)');
>>o.check('maxlag',24,'save','(acf pcf)');
>>o.estimate('save','(mdl est)');
>>o.forecast('maxlead',18,'probability',0.95,'save','(fct fvr)
');


>>o.run();
```

6.4 其余项目

6.4.1 时间聚合

下列函数允许将时间序列降频：

- dseries2M 将日期时间序列转化为月度时间序列；
- dseries2Q 将日期或月度时间序列转化为季度时间序列；
- dseries2S 将日期、月度或季度时间序列转化为半年度时间序列；
- dseries2Y 将日期、月度、季度或半年度时间序列转化为年度时间序列。

所有这些程序都有两个强制性的输入参数：第一个是 dseries 对象，第二个是聚合方法的名称（行、字符、数组）。第二个参数的可行。相应的值是：

- arithmetic-average：适用于增长率；
- geometric-average：适用于增长因子；
- sum：适用于流量变量；
- end-of-period：适用于存量变量。

示例

```
>>ts=dseries(rand(12,1),'2000M1')


ts is a dseries object:


       |Variable_1
```

```
2000M1 |0.55293
2000M2 |0.14228
2000M3 |0.38036
2000M4 |0.39657
2000M5 |0.57674
2000M6 |0.019402
2000M7 |0.57758
2000M8 |0.9322
2000M9 |0.10687
2000M10|0.73215
2000M11|0.97052
2000M12|0.60889

>>ds=dseries2Y(ts,'end-of-period')

ds is a dseries object:

     |Variable_1
2000Y|0.60889
```

6.4.2 利用单变量模型创建时间序列

用 `from` 命令递归扩展一个 `dseries` 对象是可行的。比如创建一个包含 ARMA(1,1) 模型模拟的 `dseries` 对象：

```
>>e=dseries(randn(100,1),'2000Q1','e','\varepsilon');
>>y=dseries(zeros(100,1),'2000Q1','y');
>>from 2000Q2 to 2024Q4 do y(t)=.9*y(t-1)+e(t)-.4*e(t-1);
>>y

y is a dseries object:

      |y
2000Q1|0
2000Q2|-0.95221
2000Q3|-0.6294
2000Q4|-1.8935
```

```
2001Q1|-1.1536
2001Q2|-1.5905
2001Q3|0.97056
2001Q4|1.1409
2002Q1|-1.9255
2002Q2|-0.29287
      |
2022Q2|-1.4683
2022Q3|-1.3758
2022Q4|-1.2218
2023Q1|-0.98145
2023Q2|-0.96542
2023Q3|-0.23203
2023Q4|-0.34404
2024Q1|1.4606
2024Q2|0.901
2024Q3|2.4906
2024Q4|0.79661
```

紧跟关键词 do 的表达式可以是任意的单变量方程，唯一的约束是模型不能有超前项。可以是静态模型，或者是特别的带有任意滞后阶数的非线性后向方程。命令 from 必须紧跟一个时间范围，时间范围通过 do 命令和要估计的递归表达式隔开。

7 报告

Dynare 为创建 LaTeX 报告提供了一个简单的界面，由 LaTeX 表和 PGFPLOTS/TikZ 图表组成。您可以使用通过 Dynare 创建的报告，或者挑选出您想要包含在您自己的论文中的部分（表格和图表）。尽管 Dynare 提供了通过 PGFPLOTS/TikZ 可用的选项子集，但您可以使用 PGFPLOTS/TikZ 手册中提供的选项轻松修改由 Dynare 创建的图形。您可以手动执行此操作，也可以将选项传递给 *miscTikzAxisOptions* 或 graphMiscTikzAddPlotOptions。

报告是通过调用类对象的方法来创建和修改的。对象是分层的，具有以下顺序（从高到低）：Report、Page、Section、Graph/Table/Vspace、Series。为简化语法，我们从这些类中抽象出来，允许您直接对 Report 对象进行操作，同时在您将使用的 Report 类方法中保留这些类的名称。

报告是按命令顺序创建的，因此命令的顺序很重要。当插入某个层次结构的对象时，所有方法都将在该对象上运行，直到添加了相同或更高层次结构的对象。因此，一旦您向报告添加一个 Page 后，您每次添加一个 Section 对象时，它都会被添加到这个 Page，直到另一个 Page 被添加到报告中（通过 *addPage*）。这将通过本节末尾的示例变得更加清晰。

方法选项的传递方式与 Dynare 命令的传递方式不同。它们采用 MATLAB 函数的命名选项形式，其中参数成对出现（例如 function_name(`option_1_name',`option_1_value',`option_2_name',`option_2_value',...)，其中 option_X_name 是选项的名称，而 option_X_value 是分配给该选项的值）。选项对的顺序仅在一个选项被提供两次（可能是错误的）的异常情况下才有意义。在这种情况下，传递的最后一个值是使用的值。

下面，您将看到可用于报告类的方法列表和一个说明性示例。

Constructor:report()

实例化一个 Report 对象。

选项

compiler,FILENAME：系统上 LaTeX 编译器的完整路径。如果未提供此选项，Dynare 将尝试找到合适的程序来在您的系统上编译 LaTeX。默认值取决于系统：

- Windows：findtexmf--file-type=exe pdflatex 的结果；
- macOS 和 Linux：pdflatex 的结果。

directory,FILENAME：要在其中创建报告的目录的路径。默认值：当前目录。

showDate,BOOLEAN：显示编译报告的日期和时间。默认值：true。

fileName,FILENAME：保存此报告时使用的文件名。默认值：report.tex。

header,STRING：有效的 LaTeX 在\begin{document}之前包含在报告中的代码。

默认值：empty。

maketoc,BOOLEAN：是否制作目录。每页包含一个标题一个条目。默认值：false。

margin,DOUBLE：页边距。默认值：2.5。

marginUnit,`cm'|`in'：与边距相关的单位。默认值：“cm”。

orientation,`landscape'|`portrait'：纸张方向。默认值：“纵向”。

paper,`a4'|`letter'：纸张大小。默认值：“a4”。

reportDirName,FILENAME：用于存储报告组成部分（序言、文档、结尾）的文件夹的名称。默认值：tmpRepDir。

showDate,BOOLEAN：显示编译报告的日期和时间。默认值：true。

showOutput,BOOLEAN：将报告创建进度打印到屏幕。显示创建和写入时的页码。这对于查看报告创建过程中发生潜在错误的位置很有用。默认值：true。

title,STRING：报告标题。默认值：none。

Method:addPage()

向报告添加页面。

选项

footnote,STRING：将包含在本页底部的脚注。默认值：none。

latex,STRING：用于此页面的有效 LaTeX 代码。允许用户通过直接传递 LaTeX 代码来创建要包含在报告中的页面。如果此选项被传递，页面本身将以 page_X.tex 形式保存在 *pageDirName* 目录中，其中 X 指的是页码。默认值：empty。

orientation,`landscape'|`portrait'：参见 *orientation*。

pageDirName,FILENAME：用于存储此页面的文件夹的名称。给出的目录是相对于报告类的目录选项。仅在传递 *latex* 命令时使用。默认值：tmpRepDir。

paper,`a4'|`letter'：参见 *paper*。

title,STRING|CELL_ARRAY_STRINGS：一个条目（STRING）时页面的标题。有多个条目（CELL_ARRAY_STRINGS）时，页面的标题和副标题。传递的值必须是有效的 LaTeX 代码（例如，%必须是\%）。默认值：none。

titleFormat,STRING|CELL_ARRAY_STRINGS：一个字符串，表示在 title 上使用的有效 LaTeX 标记。如果您不想使用标题（和副标题）的默认值，则元胞数组条目的数量必须等于 title 选项的数量。默认值：\large\bfseries。

titleTruncate,INTEGER：当自动生成太长的页面标题时很有用，titleTruncate 可用于在传递指定数量字符时截断标题（和后续副标题）。默认值：off。

Method:addSection()

向 Page 添加 Section。

选项

cols,INTEGER：部分中的列数。默认值：1。

height,STRING：与\sectionheight LaTeX 命令使用的字符串。默认值：'!'。

Method:addGraph()

向 Section 添加 Graph。

选项

data,dseries：为图形提供数据的 dseries。默认值：none。

axisShape,`box'|`L'：轴应具有的形状。`box'表示在图形线的左侧、右侧、底部和顶部有一条轴线。'L'``表示图形线的左侧和底部有一个轴。默认值：`box'。

graphDirName,FILENAME：用于存储此图窗的文件夹的名称。给出的目录是相对于报告类的目录选项。默认值：tmpRepDir。

graphName,STRING：保存此图窗时使用的名称。默认值：某种形式的 graph_pg1_sec2_row1_col3.tex。

height,DOUBLE：图形的高度，单位为英寸。默认值：4.5。

showGrid,BOOLEAN：是否在图形上显示主格。默认值：true。

showLegend,BOOLEAN：是否显示图例。除非您使用 graphLegendName 选项，否则图例中显示的名称是与 dseries 关联的 tex 名称。您可以使用 tex_rename 修改此 tex 名称。默认值：false。

legendAt,NUMERICAL_VECTOR：图例位置的坐标。如果此选项被传递，它会覆盖 legendLocation 选项。大小必须为 2。默认值：empty。

showLegendBox,BOOLEAN：是否在图例周围显示一个框。默认值：false。

legendLocation,OPTION：图例中图例的放置位置。OPTION 的可行值是：

```
`south west'|`south east'|`north west'|`north east'|`outer north east'
```

默认值：`south east'。

legendOrientation,`vertical'|`horizontal'：图例方向。默认值：`horizontal'。

legendFontSize,OPTION：图例条目的字体大小。OPTION 的可行值是：

```
`tiny'|`scriptsize'|`footnotesize'|`small'|`normalsize'|`large'|`Large'|`LARGE'|`huge'|`Huge'
```

默认值：tiny。

miscTikzAxisOptions,STRING：如果您对 PGFPLOTS/TikZ 感到满意，则可以使用此选项将参数直接传递给 PGFPLOTS/TikZ 轴环境命令。专门用于尚未纳入 Dynare 报告的所需 PGFPLOTS/TikZ 选项。默认值：empty。

miscTikzPictureOptions,STRING：如果您对 PGFPLOTS/TikZ 感到满意，您

可以使用此选项将参数直接传递给 PGFPLOTS/TikZ tikzpicture 环境命令。(例如，要在 x 和 y 维度上缩放图形，您可以将以下内容传递给此选项："xscale=2.5，yscale=0.5")。专门用于尚未纳入 Dynare 报告的所需``PGFPLOTS/TikZ 选项。默认值：empty。

seriesToUse,CELL_ARRAY_STRINGS：提供给 *data* 选项的 dseries 中包含的 series 的名称。如果为空，则使用提供给数据选项的所有系列。默认值：empty。

shade,dates：显示图形中应加阴影的部分的日期范围。默认值：none。

shadeColor,STRING：图形阴影部分使用的颜色。定义供 PGFPLOTS/TikZ 使用的所有有效颜色字符串都是有效的。定义的颜色列表是：

```
'red','green','blue','cyan','magenta','yellow','black','gray
','white','darkgray','lightgray','brown','lime','olive','ora
nge','pink','purple','teal','violet'.
```

此外，您可以使用这些颜色的组合。例如，如果您想要一种 20%绿色和 80%紫色的颜色，您可以传递字符串'green!20!purple'。您还可以使用 RGB 颜色，遵循以下语法：`rgb,255:red,231;green,84;blue,121'对应于 RGB 颜色（231;84;121)。PGFPLOTS/TikZ 手册修订版 1.10 的第 4.7.5 节提供了更多示例。默认值：`green'。

shadeOpacity,DOUBLE：阴影区域的不透明度必须在[0,100]内。默认值：20。

tickFontSize,OPTION：x 轴和 y 轴刻度标签的字体大小。OPTION 的可行值是：

```
`tiny'|`scriptsize' `footnotesize'|`small'|`normalsize'|`lar
ge'|`Large'|`LARGE'|`huge'|`Huge'
```

默认值：normalsize。

title,STRING|CELL_ARRAY_STRINGS：与 *title* 相同，仅用于图表。

titleFontSize,OPTION：标题的字体大小。OPTION 的可行值是：

```
`tiny'|`scriptsize' `footnotesize'|`small'|`normalsize'|`lar
ge'|`Large'|`LARGE'|`huge'|`Huge'
```

默认值：normalsize。

titleFormat,STRING：用于图形标题的格式。与 *titleFormat* 不同，由于 TikZ 的限制，此格式适用于标题和副标题。默认值：TikZ 默认。

width,DOUBLE：图形的宽度，以英寸为单位。默认值：6.0。

writeCSV,BOOLEAN：是否只用绘制的数据写入 CSV 文件。该文件将保存在由 *graphDirName* 指定的目录中，其基本名称与由 *graphName* 指定的相同，并以.csv 结尾。默认值：false。

xlabel,STRING：x 轴标签。默认值：none。

ylabel,STRING：y 轴标签。默认值：none。

xAxisTight,BOOLEAN：使用紧的 x 轴。如果为 false，则使用 PGFPLOTS/TikZ

enlarge y limits 以选择合适的轴大小。默认值：true。

xrange,dates：要在图形中显示的 x 轴上的边界。默认值：all。

xTicks,NUMERICAL_VECTOR：仅与 xTickLabels 结合使用，此选项表示标签沿 x 轴的数字位置。位置从 1 开始。默认值：与 dseries 的第一个和最后一个日期相关的索引，如果通过，则与 *shade* 选项的第一个日期相关的索引。

xTickLabels,CELL_ARRAY_STRINGS|`ALL'：要映射到 xTicks 提供的刻度的标签。默认值：dseries 的第一个和最后一个日期，如果通过，则为 *shade* 选项的第一个日期。

xTickLabelAnchor,STRIN：锚定 x 刻度标签的位置。默认值：`east'。

xTickLabelRotation,DOUBLE：旋转 x 刻度标签的量。默认值：0。

yAxisTight,BOOLEAN：使用紧的 y 轴。如果为 false，则使用 PGFPLOTS/TikZ enlarge y limits 以选择合适的轴大小。默认值：false。

yrange,NUMERICAL_VECTOR：要在图形中显示的 y 轴上的边界，表示为大小为 2 的 NUMERICAL_VECTOR，第一个条目小于第二个条目。默认值：all。

yTickLabelFixed,BOOLEAN：将 y 刻度标签四舍五入到固定的小数位数，由 yTickLabelPrecision 给出。默认值：true。

yTickLabelPrecision,INTEGER：报告 yTickLabel 的精度。默认值：0。

yTickLabelScaled,BOOLEAN：确定 y 轴是否有共同的缩放因子。默认值：true。

yTickLabelZeroFill,BOOLEAN：是否用零填充缺失的精度点。默认值：true。

showZeroline,BOOLEAN：在$y = 0$处显示黑色实线。默认值：false。

zeroLineColor,STRING：用于零线的颜色。仅在 *showZeroLine* 为真时使用。有关如何在报告中使用颜色，参见 *shadeColor* 中的说明。默认值：`black'。

Method:addTable()

向 Section 加入 Table。

选项

data,dseries：参见 *data*。

highlightRows,CELL_ARRAY_STRINGS：包含用于行突出显示的颜色的元胞数组。有关如何在报告中使用颜色，参见 *shadeColor*。可以通过使用 *addSeries* 的 *tableRowColor* 选项来覆盖特定行的突出显示。默认值：empty。

showHlines,BOOLEAN：是否显示分隔行的水平线。默认值：false。

precision,INTEGER：要在表数据中报告的小数位数（四舍五入通过离零二分法完成）。默认值：1。

range,dates：要显示的数据的日期范围。默认值：all。

seriesToUse,CELL_ARRAY_STRINGS：参见 *seriesToUse*。

tableDirName,FILENAME：用于存储此表的文件夹的名称。给出的目录是相对于报告类的目录选项。默认值：tmpRepDir。

tableName,STRING：保存此表时使用的名称。默认值：某种形式的 table_pg1_sec2_row1_col3.tex。

title,STRING：与 *title* 相同，仅用于表格。

titleFormat,STRING：与 *titleFormat* 相同，仅用于表格。默认值：\large。

vlineAfter,dates|CELL_ARRAY_DATES：在指定日期（或日期，如果传递的是日期元胞数组）之后显示一条垂直线。默认值：empty。

vlineAfterEndOfPeriod,BOOLEAN：在每个时期结束后（即每年之后、第四季度之后等）显示一条垂直线。默认值：false。

showVlines,BOOLEAN：是否显示分隔列的垂直线。默认值：false。

writeCSV,BOOLEAN：是否写入包含表中显示数据的 CSV 文件。该文件将保存在 tableDirName 指定的目录中，其基本名称与 *tableName* 指定的名称相同，结尾为.csv。默认值：false。

Method:addSeries()

向 Graph 或 Table 添加 Series。特定于图形的选项以“graph”开头，而特定于表格的选项以“table”开头。

选项

data,dseries：参见 *data*。

graphBar,BOOLEAN：是否将此系列显示为条形图，而不是默认显示为折线图。默认值：false。

graphFanShadeColor,STRING：图表中一个系列和先前添加的系列之间使用的阴影颜色。制作扇形图很有用。默认值：empty。

graphFanShadeOpacity,INTEGER：*graphFanShadeColor* 中传递的颜色不透明度。默认值：50。

graphBarColor,STRING：条形图中每个条形的轮廓颜色。只有通过 *graphBar* 才有效。默认值：`black'。

graphBarFillColor,STRING：条形图中每个条形的填充颜色。只有通过 *graphBar* 才有效。默认值：`black'。

graphBarWidth,DOUBLE：条形图中每个条形的宽度。只有通过 *graphBar* 才有效。默认值：2。

graphHline,DOUBLE：使用此选项可在给定值处绘制一条水平线。默认值：empty。

graphLegendName,STRING：本系列图例中显示的名称，作为有效的 LaTeX 传递（例如 GDP_{US},α,\color{red}GDP\color{black}）。仅当 data 和 *sho*

wLegend 选项已通过时才会显示。默认值：系列的 tex 名称。

graphLineColor,STRING：用于图表中系列的颜色。有关如何在报告中使用颜色，参见 *shadeColor* 中的说明。默认值：`black'。

graphLineStyle,OPTION：图中系列的线型。OPTION 的可行值是：

```
`none'|`solid'|`dotted'|`densely dotted'|`loosely dotted'|`dashed'|`densely dashed'|`loosely dashed'|`dashdotted'|`densely dashdotted'|`loosely dashdotted'|`dashdotdotted'|`densely dashdotdotted'|`loosely dashdotdotted'
```

默认值：`solid'。

graphLineWidth DOUBLE：图中系列的线宽。默认值：0.5。

graphMarker,OPTION：在图中系列上使用的标记。OPTION 的可行值是：

```
`x'|`+'|`-'|`|'|`o'|`asterisk'|`star'|`10-pointed star'|`oplus'|`oplus*'|`otimes'|`otimes*'|`square'|`square*'|`triangle'|`triangle*'|`diamond'|`diamond*'|`halfdiamond*'|
`halfsquare*'|`halfsquare right*'|`halfsquare left*'|`Mercedes star'|
`Mercedes star flipped'|`halfcircle'|`halfcircle*'|`pentagon'|`pentagon star'
```

默认值：none。

graphMarkerEdgeColor,STRING：图形标记的边缘颜色。如何在报告中使用颜色，参见 *shadeColor* 中的说明。默认值：graphLineColor。

graphMarkerFaceColor,STRING：图形标记的颜色。有关如何在报表中使用颜色，参见 *shadeColor* 中的说明。默认值：graphLineColor。

graphMarkerSize,DOUBLE：图形标记的大小。默认值：1。

graphMiscTikzAddPlotOptions,STRING：如果您对 PGFPLOTS/TikZ 感到满意，您可以使用此选项将参数直接传递给 PGFPLOTS/TikZaddPlots 命令。（例如，您可以将如下字符串传递给此选项，而不是传递上面的标记选项：`mark=halfcircle*,markoptions={rotate=90,scale=3}')。专门用于未纳入 Dynare 报告的所需 PGFPLOTS/TikZ 选项。默认值：empty。

graphShowInLegend,BOOLEAN：是否在图例中显示这个系列，给定 *showLegend* 选项传递给 *addGraph*。默认值：true。

graphVline,dates：使用此选项在给定日期绘制一条垂直线。默认值：empty。

tableDataRhs,dseries：要添加到当前系列右侧的系列。用于显示系列的汇总数据。例如，如果系列是季度，tableDataRhs 可以指向季度系列的年度平均值。这将导致

显示季度数据，然后是年度数据。默认值：empty。

tableRowColor,STRING：您希望该行的颜色。预定义的值包括 LightCyan 和 Gray。默认值：white。

tableRowIndent,INTEGER：表中系列名称的缩进次数。用于创建系列的子组。默认值：0。

tableShowMarkers,BOOLEAN：在表格中，如果为 true，则将每个单元格用括号括起来，并根据 tableNegColor 和 tablePosColor 对其进行着色。对图形无效。默认值：false。

tableAlignRight,BOOLEAN：是否将系列名称与单元格右侧对齐。默认值：false。

tableMarkerLimit,DOUBLE：对于小于-1*tableMarkerLimit 的值，用 tableNegColor 表示的颜色标记单元格。对于大于 tableMarkerLimit 的值，用 tablePosColor 表示的颜色标记单元格。默认值：1e-4。

tableNaNSymb,STRING：用此选项中的文本替换 NaN 值。默认值：NaN。

tableNegColor,LATEX_COLOR：标记小于零的表数据时使用的颜色。默认值：`red'。

tablePrecision,INTEGER：表数据中要报告的小数位数。默认值：precision 设置的值。

tablePosColor,LATEX_COLOR：标记大于零的表数据时使用的颜色。默认值：`blue'。

tableSubSectionHeader,STRING：表格的一个小节的标题。不会有任何数据与之关联。相当于添加了一个带有名称的空系列。默认值：''。

zeroTol,DOUBLE：零容差。任何小于 zeroTol 和大于-zeroTol 的内容都将在绘制或写入表格之前设置为零。默认值：1e-6。

Method:addParagraph()

向 Section 加入 Paragraph。

Section 只能由 Paragraphs 组成，并且必须只有 1 列。

选项

balancedCols,BOOLEAN：确定当 Paragraph 有多于一列时文本是否均匀分布在各列中。默认值：true。

cols,INTEGER：Paragraph 的列数。默认值：1.

heading,STRING：Paragraph 的标题（如节标题）。该字符串必须是有效的 LaTeX 代码。默认值：empty。

indent,BOOLEAN：是否缩进段落。默认值：true。

text,STRING：段落本身。该字符串必须是有效的 LaTeX 代码。默认值：empty。

Method:addVspace()

加入 Vspace（垂直空间）向 Section.

选项

hline,INTEGER：要插入的水平线数。默认值：0。

number,INTEGER：要插入的新行数。默认值：1。

Method:write()

写入此 Report 的 LaTeX 展示，将其保存到 *filename* 指定的文件中。

Method:compile()

将 write 编写的报告编译成 pdf 文件。如果报告尚未写入（由 filename 指定的文件的存在确定），则调用 write。

选项

compiler,FILENAME：除了不会覆盖 report 对象中包含的 compiler 的值,和 *compiler* 一样。因此，在此处传递值对于使用不同的 LaTeX 编译器或仅用于在最后一分钟传递值非常有用。

showOutput,BOOLEAN：将编译器输出打印到屏幕上。如果出现问题，LaTeX 编译器会挂起，因此对于调试代码很有用。默认值：showOutput 的值。

showReport,BOOLEAN：打开编译后的报告（适用于 Windows 和 macOS 上的 MATLAB）。默认值：true。

示例

以下代码创建一页报告。页面的第一部分包含跨两列和一行显示的两个图形。页面底部显示一个居中的表格：

```
%%Create dseries
dsq=dseries(`quarterly.csv');
dsa=dseries(`annual.csv');
dsca=dseries(`annual_control.csv');

%%Report
rep=report();

%%Page 1
rep.addPage('title',{'My Page Title','My Page Subtitle'},...
            'titleFormat',{'\large\bfseries','\large'});
```

```
%Section 1
rep.addSection('cols',2);

rep.addGraph('title','Graph Column 1','showLegend',true,...
             'xrange',dates('2007q1'):dates('2013q4'),...
             'shade',dates('2012q2'):dates('2013q4'));
rep.addSeries('data',dsq{'GROWTH_US'},'graphLineColor','blue
',... 'graphLineStyle','loosely dashed','graphLineWidth',1);
rep.addSeries('data',dsq{'GROWTH_EU'},'graphLineColor','gree
n',... 'graphLineWidth',1.5);

rep.addGraph('title','Graph Column 2','showLegend',true,...
            'xrange',dates('2007q1'):dates('2013q4'),...
            'shade',dates('2012q2'):dates('2013q4'));
rep.addSeries('data',dsq{'GROWTH_JA'},'graphLineColor','blue
',...'graphLineWidth',1);
rep.addSeries('data',dsq{'GROWTH_RC6'},'graphLineColor','gre
en',...'graphLineStyle','dashdotdotted','graphLineWidth',1.
5);

%Section 2
rep.addVspace('number',15);
rep.addSection();
rep.addTable('title','Table 1','range',dates('2012Y'):dates
('2014Y'));
shortNames={'US','EU'};
longNames={'United States','Euro Area'};
for i=1:length(shortNames)
    rep.addSeries('data',dsa{['GROWTH_' shortNames{i}]});
    delta=dsa{['GROWTH_' shortNames{i}]}-dsca{['GROWTH_' sho
rtNames{i}]};
    delta.tex_rename_('$\Delta$');
    rep.addSeries('data',delta,...'tableShowMarkers',true,'t
ableAlignRight',true);
```

```
end


%%Write & Compile Report
rep.write();
rep.compile();
```

编译后，报告如下所示：

My Page Title

My Page Subtitle

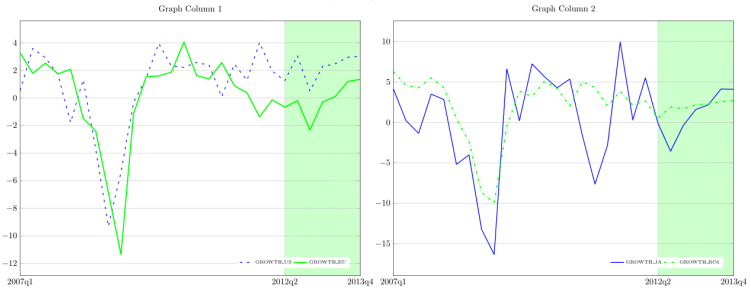

Table 1

	2012	2013	2014
GROWTH_US	1.7	2.7	3.3
Δ	[0.0]	[-0.1]	[0.0]
GROWTH_EU	-0.8	0.6	1.5
Δ	[-0.2]	[-0.2]	[-0.1]

8 示例

Dynare 附带了一个 `example.mod` 文件的数据库，演示 Dynare 多样的功能，这些大多数取自学术论文，在分配的 `example` 子文件夹中拥有这些文件。以下是一个所含例子的简短列表。更完整的描述参见文件本身的注释。

`ramst.mod`：基本实际经济周期（RBC）模型，在确定性设置中模拟；

`example1.mod`、`example2.mod`：Collard（2001）提出的两个随机性设置的小型 RBC 模型的例子（参见 Dynare 的 `guide.pdf`）；

`example3.mod`：Collard（2001）提出的一个随机性设置下的小型 RBC 模型。稳态是由 *`steady_state_model`* 模块解析求得的（参见 *`steady_state_model`*）；

`fs2000.mod`：SchorfHeId（2000）估计的现金先行模型，用于演示使用 Dynare 估计；

`fs2000_nonstationary.mod`：和 `fs2000.mod` 模型相似，但以非平稳形式写出。方程去趋势是由 Dynare 来完成的；

`bkk.mod`：Backus、Kehoe 和 Kydland（1992）提出的建立带有时间的多国 RBC 模型，用于展示如何使用 Dynare 的宏处理器；

`agtrend.mod`：Aguiar 和 Gopina（2004）提出的带有增长趋势冲击的小型开放经济 RBC 模型；

`Gali_2015.mod`：Gali（2015）的基准新凯恩斯模型，第三章中展示了如何使用 Andrle 和 Plasil（2018）中提出的“系统事前”类型的事前约束以及运行事前/事后一函数；

`NK_baseline.mod`：Fernández-Villaverde（2010）估计的基准新凯恩斯模型，说明了如何使用显性稳定状态文件更新参数并调用数值求解器；

`Occbin_example.mod`：带有两个偶然受限约束的 RBC 模型，说明如何设置 Occbin；

`Ramsey_Example.mod`：展示了在承诺（Ramsey）或使用最优简单规则（OSR）的情况下，如何在一个简单的新凯恩斯主义模型中实施最优政策实验的文件；

`Ramsey_steady_file.mod`：展示了在带有用户定义的条件稳态承诺（Ramsey）的情况下，如何在一个简单的新凯恩斯主义模型中进行最优政策实验的文件。

9 Dynare 的 misc 命令

Command:prior_function(OPTIONS);

对先验分布的参数绘制执行用户定义的函数。Dynare 返回名为 oo_.prior_function_results 的$ndraws$乘n的元胞数组中所有绘图的计算结果。

选项

function=FUNCTION_NAME：该函数必须具有以下标题 output_cell=FILENAME（xparam1、M_,options_、oo_、estim_params_、bayestopt_、dataset_、dataset_info），对所有 Dynare 结构提供只读访问。唯一允许的输出参数是$1 \times n$的元胞数组，它允许存储任何类型的输出/计算。此选项是必选的。

sampling_draws=INTEGER：用于采样的绘图数。默认值：500。

Command:posterior_function(OPTIONS);

与 *prior_function* 相同，但是属于后验分布。返回结果到 oo_.posterior_function_results。

选项

function=FUNCTION_NAME：参见 *prior_function_function*。

sampling_draws=INTEGER：参见 *prior_function_sampling_draws*。

Command:generate_trace_plots(CHAIN_NUMBER);

在指定的马尔科夫链 CHAIN_NUMBER 中，为所有估计参数和后验密度生成 MCMC 绘制的跟踪图。

MATLAB/Octave command:internals FLAG ROUTINENAME[.m]|MODFILENAME

根据 FLAG 的值，internals 命令可以运行特定 Matlab/Octave 例程（如果可用）的单元测试，显示关 Matlab/Octave 例程的文档，或者提取关于 Dynare 状态的一些信息。

标记

--test：执行与 ROUTINENAME 关联的单一测试（如果这个例程存在，并且如果 Matlab/Octave 的*.m 文件具有单一测试部分）。

示例

```
>>internals --test ROUTINENAME
```

如果是 routine.m 不在当前目录中，则必须给出完整路径：

```
>>internals --test ../matlab/fr/ROUTINENAME
```

--info：在屏幕上打印 ROUTINENAME 的内部文档（如果这个例程存在并且有一个 texinfo 内部文档头）。如果例程不在当前目录中，必须提供 ROUTINENAME 的路径。

示例

```
>>internals --doc ../matlab/fr/ROUTINENAME
```

这时只适用于少量的例程。在可用的 Matlab/Octave 程序的顶部，内部文档的注释模块

被写入 GNU texinfo 文档格式。通过调用来自 Matlab 的 texinfo 处理该模块。因此，必须在机器上安装 texinfo。

`--display-mh-history`：显示由 MODFILENAME 生成的`*.mod` 文件生成的先前保存的 MCMC 绘图的信息。此文件必须在当前目录中。

示例

```
>>internals --display-mh-history MODFILENAME
```

`--load-mh-history`：加载到 Matlab/Octave 的工作空间信息中，这些信息是涉及先前保存的，由名为 MODFILENAME 的`*.mod` 文件生成的 MCMC 绘图。

示例

```
>>internals --load-mh-history MODFILENAME
```

创建一个名为 `mcmc_informations` 的结构（在工作区中），具有以下字段：

- `Nblck`：MCMC 链的数目；
- `InitialParameters`：`Nblck*n`，其中 `n` 是估计参数的数目，双精度数组。MCMC 的初始状态；
- `LastParameters`：`Nblck*n`，其中 `n` 是估计参数的数目，双精度数组。MCMC 的当前状态；
- `InitialLogPost`：`Nblck*1` 双精度数组。后验核的初始值；
- `LastLogPost`：`Nblck*1` 双精度数组。后验核的当前值；
- `InitialSeeds`：`1*Nblck` 结构数组。随机数发生器的初始状态；
- `LastSeeds`：`1*Nblck` 结构数组。随机数发生器的当前状态；
- `AcceptanceRatio`：`1*Nblck` 双精度数组。目前接受的比率。

MATLAB/Octave command:prior`[OPTIONS[,...]];`

根据选项打印关于先验分布的各种信息。如果没有提供选项，则命令返回可用选项列表。以下选项可用：

选项

`table`：建立一个表格描述边际收入分配情况（平均值、众数、标准差、上下限、HPD 区间）。

`moments`：计算并显示先验众数的内生变量的一阶、二阶矩（考虑模型线性化版本）。

`moments(distribution)`：通过从先验中随机抽样，计算并显示内生变量（考虑模型的线性化版本）的一阶矩和二阶矩的先验均值和先验标准差，结果也将存储在 `prior` 子文件夹中的`_endogenous_variables_prior_drawn.mat` 文件。

`optimize`：优化先验密度（从随机初始猜测开始）。使稳态不存在或不满足 Blanchard 和 Kahn 条件的参数处于不利状态，就像它们在最大化后验密度时一样。如果在这些区域上定义了显著比例的先验质量，则优化算法可能无法收敛到真实解（先验模式）。

simulate：使用蒙特卡洛计算有效先验质量。理想情况下，有效先验质量应该等于1，否则当最大化后验密度时可能出现问题，并且基于边际密度的模型比较可能不公平。当比较模型如A和B时，对于估计的有效先验质量$p_A \neq p_B \leq 1$，边际密度m_A和m_B应该被校正，以便比较模型的先验质量是相同的。

plot：绘制边际先验密度。

10 参考文献

- Abramowitz, Milton and Irene A. Stegun(1964): “Handbook of Mathematical Functions”, Co urier Dover Publications.
- Adjemian, Stéphane, Matthieu Darracq Parriès and Stéphane Moyen(2008): “Towards a mon etary policy evaluation framework”, *European Central Bank Working Paper*, 942.
- Aguiar, Mark and Gopinath, Gita(2004): “Emerging Market Business Cycles: The Cycle is th e Trend,” *NBER* Working Paper, 10734.
- Amisano, Gianni and Tristani, Oreste(2010): “Euro area inflation persistence in an estimated nonlinear DSGE model”, *Journal of Economic Dynamics and Control*, 34(10), 1837–1858.
- Andreasen, Martin M., Jesús Fernández-Villaverde, and Juan Rubio-Ramírez(2018): “The Pr uned State-Space System for Non-Linear DSGE Models: Theory and Empirical Applications,” *Review of Economic Studies*, 85(1), pp. 1-49.
- Andrews, Donald W.K(1991): “Heteroskedasticity and autocorrelation consistent covariance matrix estimation”, *Econometrica*, 59(3), 817–858.
- Backus, David K., Patrick J. Kehoe, and Finn E. Kydland(1992): “International Real Busines s Cycles,” *Journal of Political Economy*, 100(4), 745–775.
- Baxter, Marianne and Robert G. King(1999): “Measuring Business Cycles: Approximate Ban d-pass Filters for Economic Time Series,” *Review of Economics and Statistics*, 81(4), 575–59 3.
- Born, Benjamin and Johannes Pfeifer(2014): “Policy risk and the business cycle”, *Journal of Monetary Economics*, 68, 68-85
- Boucekkine, Raouf(1995): “An alternative methodology for solving nonlinear forward-lookin g models,” *Journal of Economic Dynamics and Control*, 19, 711–734.
- Brayton, Flint and Peter Tinsley(1996): “A Guide to FRB/US: A Macroeconomic Model of th e United States”, *Finance and Economics Discussion Series*, 1996-42.
- Brayton, Flint, Morris Davis and Peter Tulip(2000): “Polynomial Adjustment Costs in FRB/ US”, *Unpublished manuscript*.
- Brooks, Stephen P., and Andrew Gelman(1998): “General methods for monitoring convergen ce of iterative simulations,” *Journal of Computational and Graphical Statistics*, 7, pp. 434–4 55.
- Cardoso, Margarida F., R. L. Salcedo and S. Feyo de Azevedo(1996): “The simplex simulate d annealing approach to continuous non-linear optimization,” *Computers & Chemical Engine ering*, 20(9), 1065-1080.
- Chib, Siddhartha and Srikanth Ramamurthy(2010): “Tailored randomized block MCMC met

hods with application to DSGE models," *Journal of Econometrics*, 155, 19–38.

- Christiano, Lawrence J., Mathias Trabandt and Karl Walentin(2011): "Introducing financial f rictions and unemployment into a small open economy model," *Journal of Economic Dynami cs and Control*, 35(12), 1999–2041.
- Christoffel, Kai, Günter Coenen and Anders Warne(2010): "Forecasting with DSGE models," *ECB Working Paper Series*, 1185.
- Collard, Fabrice(2001): "Stochastic simulations with Dynare: A practical guide".
- Collard, Fabrice and Michel Juillard(2001a): "Accuracy of stochastic perturbation methods: The case of asset pricing models," *Journal of Economic Dynamics and Control*, 25, 979–999.
- Collard, Fabrice and Michel Juillard(2001b): "A Higher-Order Taylor Expansion Approach t o Simulation of Stochastic Forward-Looking Models with an Application to a Non-Linear Ph illips Curve," *Computational Economics*, 17, 125–139.
- Corona, Angelo, M. Marchesi, Claudio Martini, and Sandro Ridella(1987): "Minimizing mult imodal functions of continuous variables with the "simulated annealing" algorithm", *ACM Tr ansactions on Mathematical Software*, 13(3), 262–280.
- Cuba-Borda, Pablo, Luca Guerrieri, Matteo Iacoviello, and Molin Zhong(2019): "Likelihood evaluation of models with occasionally binding constraints", *Journal of Applied Econometric s*, 34(7), 1073-1085
- Del Negro, Marco and Franck Schorfheide(2004): "Priors from General Equilibrium Models for VARs", *International Economic Review*, 45(2), 643–673.
- Dennis, Richard(2007): "Optimal Policy In Rational Expectations Models: New Solution Alg orithms", *Macroeconomic Dynamics*, 11(1), 31–55.
- Durbin, J. and S. J. Koopman(2012), *Time Series Analysis by State Space Methods*, Second R evised Edition, Oxford University Press.
- Fair, Ray and John Taylor(1983): "Solution and Maximum Likelihood Estimation of Dynami c Nonlinear Rational Expectation Models," *Econometrica*, 51, 1169–1185.
- Fernández-Villaverde, Jesús(2010): "The econometrics of DSGE models," *SERIEs*, 1, 3–49.
- Fernández-Villaverde, Jesús and Juan Rubio-Ramírez(2004): "Comparing Dynamic Equilibri um Economies to Data: A Bayesian Approach," *Journal of Econometrics*, 123, 153–187.
- Fernández-Villaverde, Jesús and Juan Rubio-Ramírez(2005): "Estimating Dynamic Equilibri um Economies: Linear versus Nonlinear Likelihood," *Journal of Applied Econometrics*, 20, 8 91–910.
- Ferris, Michael C. and Todd S. Munson(1999): "Interfaces to PATH 3.0: Design, Implementa tion and Usage", *Computational Optimization and Applications*, 12(1), 207–227.

- Galí, Jordi(2015): “Monetary Policy, Inflation, and the Business Cycle,” 2nd Edition, Princet on University Press, Princeton.
- Geweke, John(1992): “Evaluating the accuracy of sampling-based approaches to the calculati on of posterior moments,” in J.O. Berger, J.M. Bernardo, A.P. Dawid, and A.F.M. Smith(eds.) Proceedings of the Fourth Valencia International Meeting on Bayesian Statistics, pp. 169–19 4, Oxford University Press.
- Geweke, John(1999): “Using simulation methods for Bayesian econometric models: Inferenc e, development and communication,” *Econometric Reviews*, 18(1), 1–73.
- Giovannini, Massimo, Philipp Pfeiffer, and Marco Ratto(2021), “Efficient and robust inferen ce of models with occasionally binding constraints,” Working Papers 2021-03, Joint Researc h Centre, European Commission
- Giordani, Paolo, Michael Pitt, and Robert Kohn(2011): “Bayesian Inference for Time Series State Space Models” in: *The Oxford Handbook of Bayesian Econometrics*, ed. by John Gewe ke, Gary Koop, and Herman van Dijk, Oxford University Press, 61–124.
- Guerrieri, Luca and Matteo Iacoviello(2015): “OccBin: A toolkit for solving dynamic models with occasionally binding constraints easily,” *Journal of Monetary Economics*, 70, 22–38.
- Goffe, William L., Gary D. Ferrier, and John Rogers(1994): “Global Optimization of Statistic al Functions with Simulated Annealing,” *Journal of Econometrics*, 60(1/2), 65–100.
- Hansen, Lars P.(1982): “Large sample properties of generalized method of moments estimato rs,” Econometrica, 50(4), 1029–1054.
- Hansen, Nikolaus and Stefan Kern(2004): “Evaluating the CMA Evolution Strategy on Multi modal Test Functions”. In: *Eighth International Conference on Parallel Problem Solving fro m Nature PPSN VIII*, Proceedings, Berlin: Springer, 282–291.
- Harvey, Andrew C. and Garry D.A. Phillips(1979): “Maximum likelihood estimation of regre ssion models with autoregressive-moving average disturbances,” *Biometrika*, 66(1), 49–58.
- Herbst, Edward(2015): “Using the “Chandrasekhar Recursions” for Likelihood Evaluation of DSGE Models,” *Computational Economics*, 45(4), 693–705.
- Ireland, Peter(2004): “A Method for Taking Models to the Data,” *Journal of Economic Dyna mics and Control*, 28, 1205–26.
- Iskrev, Nikolay(2010): “Local identification in DSGE models,” *Journal of Monetary Econom ics*, 57(2), 189–202.
- Judd, Kenneth(1996): “Approximation, Perturbation, and Projection Methods in Economic A nalysis”, in *Handbook of Computational Economics*, ed. by Hans Amman, David Kendrick, a nd John Rust, North Holland Press, 511–585.

- Juillard, Michel(1996): “Dynare: A program for the resolution and simulation of dynamic mo dels with forward variables through the use of a relaxation algorithm,” CEPREMAP, *Couvert ure Orange*, 9602.
- Kanzow, Christian and Stefania Petra(2004): “On a semismooth least squares formulation of complementarity problems with gap reduction,” *Optimization Methods and Software*, 19, 507 –525.
- Kim, Jinill and Sunghyun Kim(2003): “Spurious welfare reversals in international business c ycle models,” *Journal of International Economics*, 60, 471–500.
- Kim, Jinill, Sunghyun Kim, Ernst Schaumburg, and Christopher A. Sims(2008): “Calculating and using second-order accurate solutions of discrete time dynamic equilibrium models,” *Jo urnal of Economic Dynamics and Control*, 32(11), 3397–3414.
- Komunjer, Ivana and Ng, Serena(2011): ”Dynamic identification of dynamic stochastic gener al equilibrium models”, *Econometrica*, 79, 1995–2032.
- Koop, Gary(2003), *Bayesian Econometrics*, John Wiley & Sons.
- Koopman, S. J. and J. Durbin(2000): “Fast Filtering and Smoothing for Multivariate State Sp ace Models,” *Journal of Time Series Analysis*, 21(3), 281–296.
- Koopman, S. J. and J. Durbin(2003): “Filtering and Smoothing of State Vector for Diffuse St ate Space Models,” *Journal of Time Series Analysis*, 24(1), 85–98.
- Kuntsevich, Alexei V. and Franz Kappel(1997): “SolvOpt - The solver for local nonlinear opt imization problems(version 1.1, Matlab, C, FORTRAN)”, University of Graz, Graz, Austria.
- Laffargue, Jean-Pierre(1990): “Résolution d’un modèle macroéconomique avec anticipations rationnelles”, *Annales d’Économie et Statistique*, 17, 97–119.
- Liu, Jane and Mike West(2001): “Combined parameter and state estimation in simulation-bas ed filtering”, in *Sequential Monte Carlo Methods in Practice*, Eds. Doucet, Freitas and Gordo n, Springer Verlag.
- Murray, Lawrence M., Emlyn M. Jones and John Parslow(2013): “On Disturbance State-Spa ce Models and the Particle Marginal Metropolis-Hastings Sampler”, *SIAM/ASA Journal on U ncertainty Quantification*, 1, 494–521.
- Mutschler, Willi(2015): “Identification of DSGE models - The effect of higher-order approxi mation and pruning“, *Journal of Economic Dynamics & Control*, 56, 34-54.
- Mutschler, Willi(2018): “Higher-order statistics for DSGE models”, *Econometrics and Statist ics*, 6(C), 44-56.
- Pearlman, Joseph, David Currie, and Paul Levine(1986): “Rational expectations models with partial information,” *Economic Modelling*, 3(2), 90–105.

- Planas, Christophe, Marco Ratto and Alessandro Rossi(2015): “Slice sampling in Bayesian es timation of DSGE models”.
- Pfeifer, Johannes(2013): “A Guide to Specifying Observation Equations for the Estimation of DSGE Models”.
- Pfeifer, Johannes(2014): “An Introduction to Graphs in Dynare”.
- Qu, Zhongjun and Tkachenko, Denis(2012): “Identification and frequency domain quasi-max imum likelihood estimation of linearized dynamic stochastic general equilibrium models“, *Q uantitative Economics*, 3, 95–132.
- Rabanal, Pau and Juan Rubio-Ramirez(2003): “Comparing New Keynesian Models of the Bu siness Cycle: A Bayesian Approach,” Federal Reserve of Atlanta, *Working Paper Series*, 200 3-30.
- Raftery, Adrian E. and Steven Lewis(1992): “How many iterations in the Gibbs sampler?,” in *Bayesian Statistics, Vol. 4*, ed. J.O. Berger, J.M. Bernardo, A.P. * Dawid, and A.F.M. Smith, Clarendon Press: Oxford, pp. 763-773.
- Ratto, Marco(2008): “Analysing DSGE models with global sensitivity analysis”, *Computatio nal Economics*, 31, 115–139.
- Ratto, Marco and Iskrev, Nikolay(2011): “Identification Analysis of DSGE Models with DY NARE.“, *MONFISPOL* 225149.
- Ruge-Murcia, Francisco J.(2012): “Estimating nonlinear DSGE models by the simulated met hod of moments: With an application to business cycles“, *Journal of Economic Dynamics an d Control*, 36, 914-938.
- Schmitt-Grohé, Stephanie and Martin Uríbe(2004): “Solving Dynamic General Equilibrium Models Using a Second-Order Approximation to the Policy Function,” *Journal of Economic Dynamics and Control*, 28(4), 755–775.
- Schnabel, Robert B. and Elizabeth Eskow(1990): “A new modified Cholesky algorithm,” *SIA M Journal of Scientific and Statistical Computing*, 11, 1136–1158.
- Schorfheide, Frank(2000): “Loss Function-based evaluation of DSGE models,” *Journal of A pplied Econometrics*, 15(6), 645–670.
- Sims, Christopher A., Daniel F. Waggoner and Tao Zha(2008): “Methods for inference in lar ge multiple-equation Markov-switching models,” *Journal of Econometrics*, 146, 255–274.
- Skoeld, Martin and Gareth O. Roberts(2003): “Density Estimation for the Metropolis-Hastin gs Algorithm,” *Scandinavian Journal of Statistics*, 30, 699–718.
- Smets, Frank and Rafael Wouters(2003): “An Estimated Dynamic Stochastic General Equilib rium Model of the Euro Area,” *Journal of the European Economic Association*, 1(5), 1123–1

175.

- Stock, James H. and Mark W. Watson(1999). “Forecasting Inflation,”, *Journal of Monetary Economics*, 44(2), 293–335.

- Uhlig, Harald(2001): “A Toolkit for Analysing Nonlinear Dynamic Stochastic Models Easily,” in *Computational Methods for the Study of Dynamic Economies*, Eds. Ramon Marimon and Andrew Scott, Oxford University Press, 30–61.
- U.S. Census Bureau(2017): “X-13 ARIMA-SEATS Reference Manual”.
- Villemot, Sébastien(2011): “Solving rational expectations models at first order: what Dynare does,” *Dynare Working Papers*, 2, CEPREMAP.

附表：字段汇总表

表 1 模型数组 M_.字段

字段	名称	字段	名称
M_.endo_names	内生变量名称	M_.npred	后向变量个数
M_.endo_names_long	内生变量名称（长字符版本）	M_.nsfwrd	内生前向变量个数
M_.endo_nbr	内生变量个数	M_.nspred	内生状态变量个数
M_.endo_partitions.NAME	内生变量名称分区	M_.nstatic	静态变量个数
M_.exo_det_names	外生确定性变量	M_.orig_endo_nbr	创建辅助变量前的内生变量数量
M_.exo_det_names_long	外生确定性变量（长字符版本）	M_.osr.param_bounds	执行 osr 规则的参数矩阵上、下限
M_.exo_det_partitions.NAME	外生确定性变量名称分区	M_.osr.param_indices	执行 osr 规则的参数名称
M_.exo_names	外生冲击名称	M_.osr.param_names	执行 osr 规则的目标函数变量索引
M_.exo_names_long	外生变量名称（长字符版本）	M_.osr.variable_indices	执行 osr 规则的目标函数变量索引
M_.exo_nbr	外生变量个数	M_.osr.variable_weights	执行 osr 规则的变量关联加权矩阵
M_.exo_partitions.NAME	外生变量名称分区	M_.param_names	参数名称
M_.H	测量误差取值	M_.param_names_long	参数名称（长字符版本）
M_.lead_lag_incidence	内生变量在动态模型雅可比行列式对应列位置的矩阵	M_.param_nbr	参数个数
M_.matched_moments	估算积矩	M_.param_partitions.NAME	参数名称分区
M_.matched_moments_orig	估算积矩（带有原始模块）	M_.params	参数取值
M_.nboth	混合变量个数	M_.Sigma_e	外生冲击的方差协方差矩阵
M_.ndynamic	动态变量个数	M_.sigma_e_is_diagonal	Sigma_e 是否为对角矩阵
M_.nfwrd	前向变量个数	M_.state_var	状态变量的索引向量

表 2 结果数组 oo_.字段

字段	名称
oo_.autocorr	内生变量的自相关矩阵
oo_.conditional_forecast	预测变量
oo_.conditional_forecast.cond	条件预测变量

oo_.conditional_forecast.controlled_exo_variables	基于条件预测的受控外生变量值
oo_.conditional_forecast.controlled_variables	受约束的内生预测变量
oo_.conditional_forecast.graphs	生成条件预测图信息的变量
oo_.conditional_forecast.uncond	无条件预测变量
oo_.conditional_shock_decomposition	冲击分解数组
oo_.conditional_variance_decomposition	方差分解数组
oo_.conditional_variance_decomposition_ME	方差分解数组（存在测量误差）
oo_.contemporaneous_correlation	同期相关性数组
oo_.convergence.geweke	Geweke 收敛数组
oo_.dr	决策规则系数数组
oo_.dr.eigval	决策规则系数数组特征值
oo_.dr.g_0	常数修正项
oo_.dr.g_1	变量的系数矩阵
oo_.dr.g_2	变量二阶泰勒展开式系数矩阵
oo_.dr.g_3	变量三阶泰勒展开式矩阵
oo_.dr.ghs2	冲击方差的转移效应
oo_.dr.ghu	外生冲击的系数矩阵
oo_.dr.ghuu	外生冲击二阶泰勒展开式系数矩阵
oo_.dr.ghx	状态变量的系数矩阵
oo_.dr.ghxu	状态变量和外生冲击交互项的系数矩阵
oo_.dr.ghxx	状态变量二阶泰勒展开式系数矩阵
oo_.dr.inv_order_var	决策规则顺序到声明顺序的逆映射
oo_.dr.order_var	决策规则顺序到声明顺序的映射
oo_.dr.state_var	状态变量的数字索引
oo_.dr.ys	内生变量的稳态值
oo_.dsge_var.posterior_mode	DSGE-VAR 模型的估计结果
oo_.endo_simul	内生变量的模拟样本
oo_.endo_val	内生变量初值、历史值和终值

oo_.exo_simul	外生冲击的模拟样本矩阵
oo_.FilterCovariance	利用 Metropolis 采样算法的预测协方差矩阵一步向前误差序列
oo_.Filtered_Variables_X_step_ahead	k 步超前滤波变量（贝叶斯估计情形）
oo_.FilteredVariables	滤波变量
oo_.FilteredVariablesKStepAhead	k 步超前滤波变量
oo_.FilteredVariablesKStepAheadVariances	k 步超前滤波变量方差
oo_.FilteredVariablesShockDecomposition	滤波变量冲击分解
oo_.forecast	预测的后验分布数组（无 MH 算法）
oo_.gamma{1}_y	方差/协方差矩阵
oo_.gamma{i+1}_y	自相关函数
oo_.gamma{ar+2}_y	无条件方差分解
oo_.gamma{ar+3}_y	均值修正项向量
oo_.initval_series	初始值序列
oo_.irfs	脉冲响应函数
oo_.kurtosis	内生变量的偏度
oo_.MarginalDensity.LaplaceApproximation	基于拉普拉斯近似的边际数据密度
oo_.MarginalDensity.ModifiedHarmonicMean	基于 Geweke 改进的调和平均估计量
oo_.mean	内生变量的理论或模拟均值
oo_.MeanForecast	平均化冲击不确定性的预测分布
oo_.Model_Comparison	模型比较的结果
oo_.mom	矩估计法的结果
oo_.mom.data_moments	所选数据经验矩的平均值
oo_.mom.gmm_stage_*_mode	存储阶段*的 GMM 估计值
oo_.mom.gmm_stage_*_std_at_mode	存储阶段*的 GMM 估计标准差
oo_.mom.J_test	矩估计的 J 统计量
oo_.mom.m_data	每个时间点的选定经验矩
oo_.mom.model_moments	隐含的选定模型矩（给定当前参数猜测）
oo_.mom.model_moments_params_derivs	经分析计算的模型矩相对于估计参数的导数的雅可比矩阵

oo_.mom.Q	二阶矩距离目标函数的标量值
oo_.mom.smm_stage_*_mode	存储阶段*的 SMM 估计值
oo_.mom.smm_stage_*_std_at_mode	存储阶段*的 SMM 估计标准差
oo_.mom.Sw	使用的加权矩阵 Cholesky 分解
oo_.mom.verbose_gmm_stage_*_mode	存储阶段*和中间估计结果的 GMM 估计值
oo_.mom.verbose_gmm_stage_*_std_at_mode	存储阶段*和中间估计结果的 GMM 估计标准差
oo_.mom.verbose_smm_stage_*_mode	存储阶段*和中间估计结果的 SMM 估计值
oo_.mom.verbose_smm_stage_*_std_at_mode	存储阶段*和中间估计结果的 SMM 估计标准差
oo_.ms	马尔科夫转换 SBVAR 模型结果
oo_.occbin	偶然受限约束的模拟结果
oo_.occbin.exo_pos	偶然受限约束的外生变量位置
oo_.occbin.simul.linear	偶然受限约束基于线性解模拟的矩阵
oo_.occbin.simul.piecewise	偶然受限约束基于分段线性解模拟的矩阵
oo_.occbin.simul.regime_history	偶然受限约束相应时期发生冲击的状态历史信息结构
oo_.occbin.simul.shocks_sequence	偶然受限约束模拟过程中使用冲击序列的矩阵
oo_.occbin.simul.ys	偶然受限约束的稳态值向量
oo_.occbin.smoother.regime_history	偶然受限约束平滑器状态历史信息结构
oo_.osr.objective_function	最优简单规则目标函数
oo_.osr.optim_params	执行 osr 规则的最优点参数值
oo_.osr.optim_params.PARAMETER_NAME	执行 osr 规则的最优点参数名称
oo_.planner_objective_value	计划者目标函数的结果
oo_.planner_objective_value.conditional	指定初始条件和零初值拉格朗日乘子的福利值标量（完美预期的背景）
oo_.planner_objective_value.conditional.steady_initial_multipliers	计划者目标设置为 0（初始拉格朗日乘数关联计划者的目标）
oo_.planner_objective_value.conditional.zero_initial_multipliers	计划者目标设置为稳态值（初始拉格朗日乘数关联计划者的目标）
oo_.planner_objective_value.unconditional	无条件福利值的标量
oo_.PointForecast	考虑到参数和冲击不确定性的预测分布
oo_.posterior.metropolis	Metropolis 采样算法的后验
oo_.posterior.optimization	后验分布最优化结果

oo_.posterior_density	后验分布密度
oo_.posterior_function_results	后验分布函数结果
oo_.posterior_hpdinf	后验分布 90%HPD 区间的下界
oo_.posterior_hpdsup	后验分布 90%HPD 区间的上界
oo_.posterior_mean	后验分布均值
oo_.posterior_median	后验分布中位数
oo_.posterior_mode	后验分布众数
oo_.posterior_std	后验分布标准差
oo_.posterior_std_at_mode	众数附近的后验分布标准差
oo_.posterior_var	后验分布方差
oo_.PosteriorIRF.dsge	DSGE 模型后验分布脉冲响应
oo_.PosteriorTheoreticalMoments	后验分布的理论矩
oo_.PosteriorTheoreticalMoments.dsge.ConditionalVarianceDecomposition	DSGE 模型条件方差分解的后验分布理论矩
oo_.PosteriorTheoreticalMoments.dsge.ConditionalVarianceDecompositionME	DSGE 模型条件方差分解的后验分布理论矩（存在测量误差）
oo_.PostiorIRF.dsge	DESG 模型后验分布脉冲响应（贝叶斯估计）
oo_.prior_function_results	先验分布函数结果
oo_.realtime_conditional_shock_decomposition	实时条件分解结果
oo_.realtime_forecast_shock_decomposition	实时预测分解结果
oo_.realtime_shock_decomposition	实时冲击分解结果
oo_.RecursiveForecast	递归预测结果
oo_.shock_decomposition	冲击分解结果
oo_.skewness	内生变量的偏度
oo_.SmoothedMeasurementErrors	测量误差的平滑值
oo_.SmoothedShocks	外生变量的平滑值
oo_.SmoothedVariables	内生变量的平滑值
oo_.Smoother.Constant	平滑器使用的内生变量常量
oo_.Smoother.loglinear	平滑器使用的对数线性
oo_.Smoother.State_uncertainty	卡尔曼平滑的完整数据状态估计的协方差矩阵

oo_.Smoother.SteadyState	平滑器使用的内生变量稳态分量
oo_.Smoother.Trend	平滑器使用的变量趋势组件
oo_.Smoother.TrendCoeffs	平滑器使用的观察变量趋势系数
oo_.Smoother.Variance	卡尔曼平滑的超前预测误差协方差矩阵
oo_.SpectralDensity	模型变量的谱密度
oo_.steady_state	计算的稳态
oo_.time	Dynare 运行的总计算时间
oo_.trend_component.MODEL_NAME.CompanionMatrix	趋势成分模型的简化形式伴随矩阵
oo_.UpdatedVariables	给定变量的更新值
oo_.var	内生变量的理论或模拟方差协方差矩阵
oo_.var.MODEL_NAME.CompanionMatrix	VAR 模型的简化形式伴随矩阵
oo_.variance_decomposition	无条件方差分解数组（存在测量误差）
oo_.variance_decomposition_ME	无条件方差分解数组
oo_recursive	执行递归估计和预测时估计不同样本的单元阵列