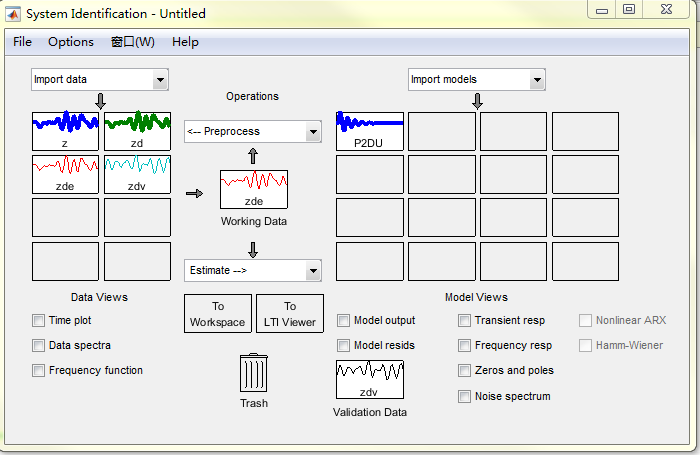
系统辨识方法总结1

驱动组 王彬 2016.11.10

# 1：matlab 系统辨识工具箱APP

Matlab系统辨识工具箱虽然具体算法已经由matlab实现，但是可以比较方便的学习系统辨识的流程和需要注意的地方。具有参考价值。

## 1.1：系统辨识App的使用简介



上图为工具箱界面，左边是数据部分，右边是模型部分。左边可以导入很多数据，生成的中间数据也会在左边显示，左边的数据可以通过拖拽的方式进行拉动到working Data区等。

Preprocess可以对辨识前的数据进行一些处理，比如去掉平均值等。Estimate是各种模型的估计，比如多项式模型，状态空间模型等。Preprocess 和 Estimate 中都有quick start，可以自动完成一部分工作。不需要的数据可以直接拖动到trash中。各个单选按钮例如Time plot可以显示波形，各个模型的输出比较图，阶跃响应，频率响应等。

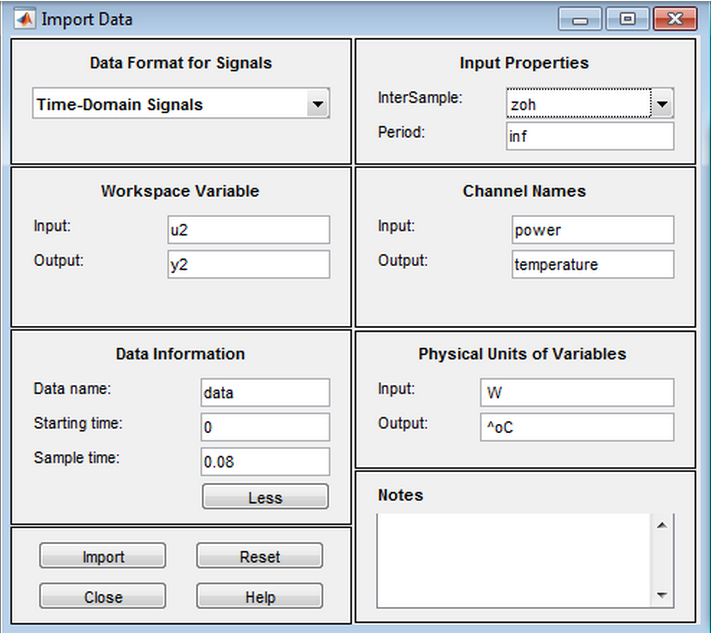
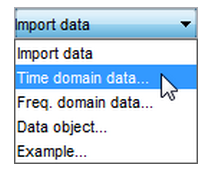
## 1.2：APP准备数据过程

可通过systemIdentification在工作空间中打开辨识APP,也可以直接通过应用程序打开

步骤如下：

（1）：输入数据

需要在工作空间上先导入数据，然后再通过这个工具导入数据

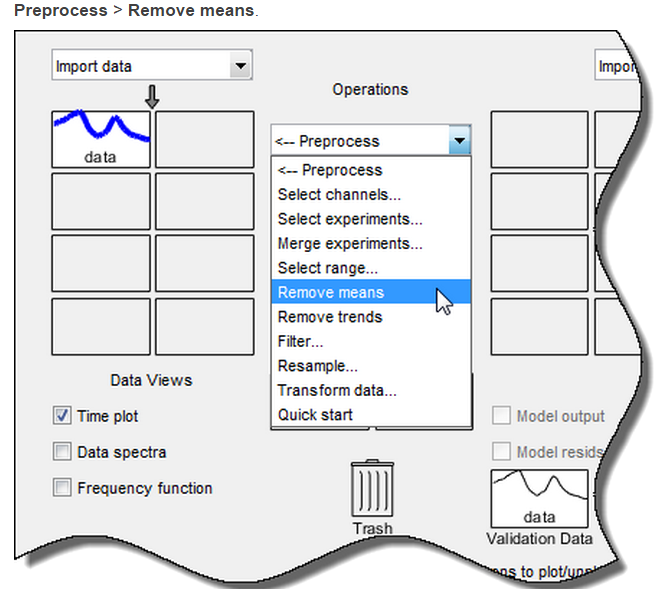


同时可设置数据开始时间，采样时间。

点击more可以显示更多信息，包括设置名称和单位等

（2）去除平均值

因为系统数据一般是在平衡点附近的值，所以可以去掉均值。相当于去掉直流偏置。

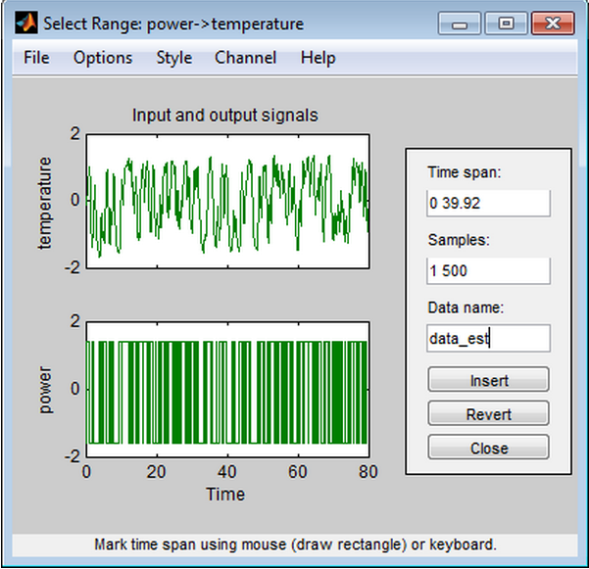
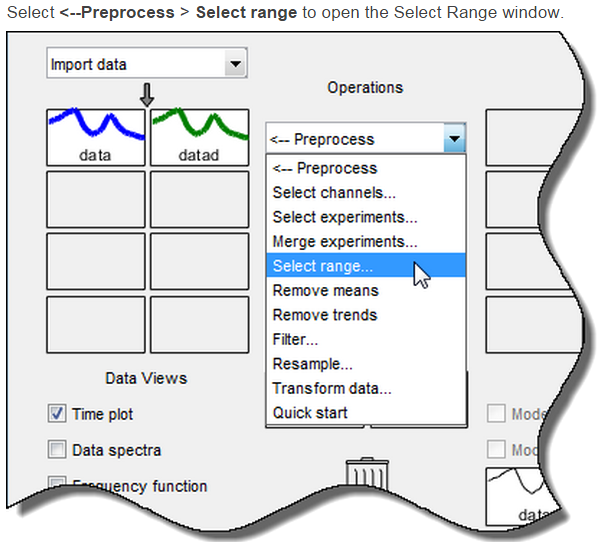


（3）：将去掉平均值的数据作为工作数据

可通过拖拽的方式将数据托到工作数据区域。

（4）数据分割

将输入的数据分割为两部分，一部分用于估计，一部分用于验证，一般都是输入一组数据，需要做这个处理，如果输入多组数据，可以将一组用于估计，一组用于验证。



上图右边samples 中就是分割数据，1 500 表示从1—500 共500个数据作为估计的数据集，点击插入即可，验证的部分同理。

（5）:将生成的估计和验证数据拖到对应的位置

通过拖拽即可。

## 1.3：APP保存session

上一步导入的数据，可以通过保存session的方式将其保存。

File > Save session as

下次可以直接打开，例如

systemIdentification('dryer2\_processed\_data')

## 1.4：APP估计线性模型quick start

可以使用Estimate > Quick start.

它会估计四种模型

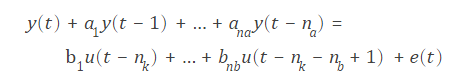
1：阶跃响应

2：频率分析

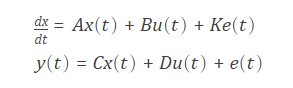
3：四阶ARX模型

4：状态空间模型

ARX模型如下



状态空间模型如下



阶跃响应一般用来估计delay时间。

模型输出用来显示不同模型的输出，和实际测量的输出之间进行比较，看哪一种模型和输出最匹配。

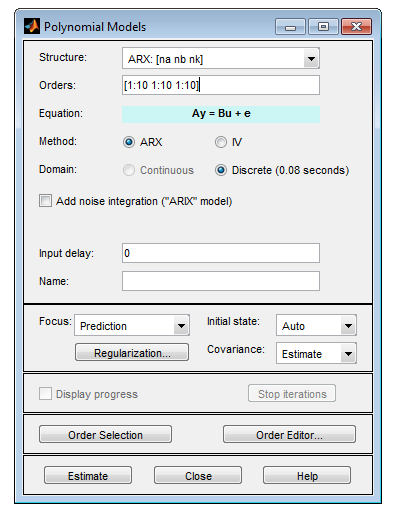
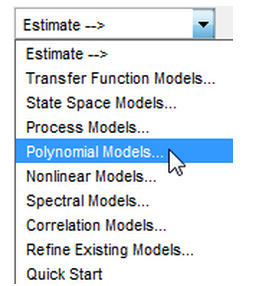
## 1.5：估计精确线性模型

对于一个黑盒子来说，必须具体模型的阶数，那么怎么才能知道阶数是否合适？需要设置一定的范围的阶数和delay等，然后看估计出的不同模型的匹配度哪个最好。

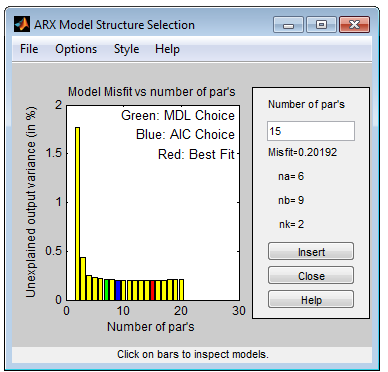
### 1.5.1：ARX模型

一般使用ARX模型，na,nb,nk分别是极点，零点，和延迟。

C:\Users\googol\AppData\Local\Temp\mx31EFE.png



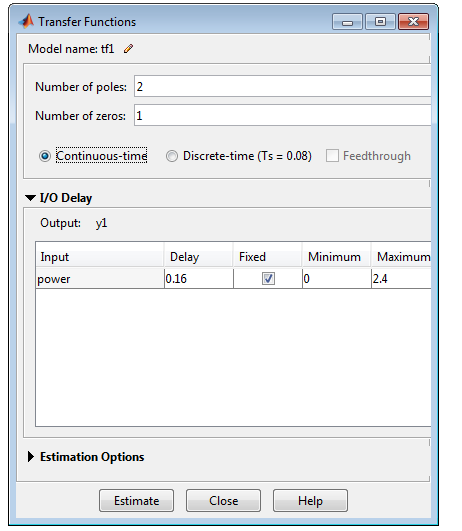
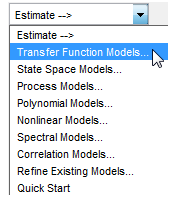
点击estimate以后，会得到哪一组的零极点和delay是最好的。标准是验证数据和模型预测数据的差的平方和最小。不同颜色代表不同的标准。选择最好的匹配点击insert就可以插入新的模型。如果从低阶到高阶都差不多，那么说明高阶并不能获取更准确的模型。



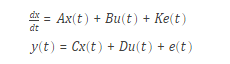
### 1.5.2：传递函数模型

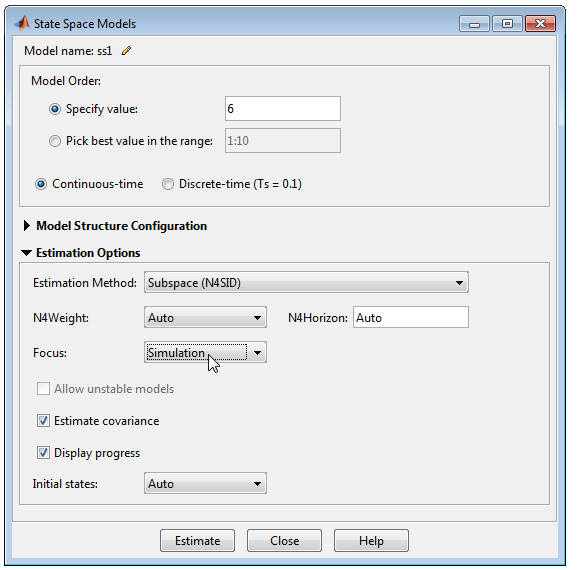
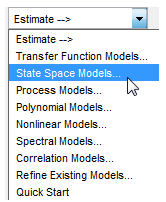
按照上面获取的比较好的参数输入到传递函数模型，可以估计出对应的传递函数模型

C:\Users\googol\AppData\Local\Temp\mx3C838.png



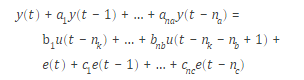
### 1.5.3：状态空间模型



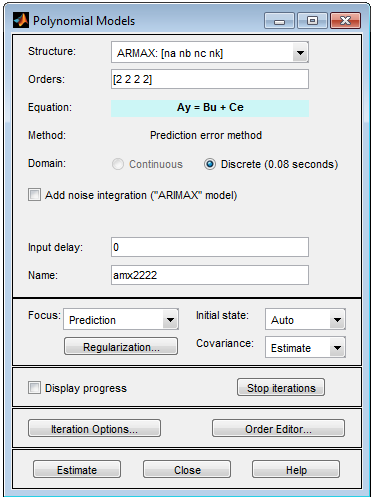


通过设置阶数，估计方法等，得到状态空间的模型。

### 1.5.4：ARMAX模型



因为该模型包含了噪声的多项式表达，所以比ARX模型更灵活

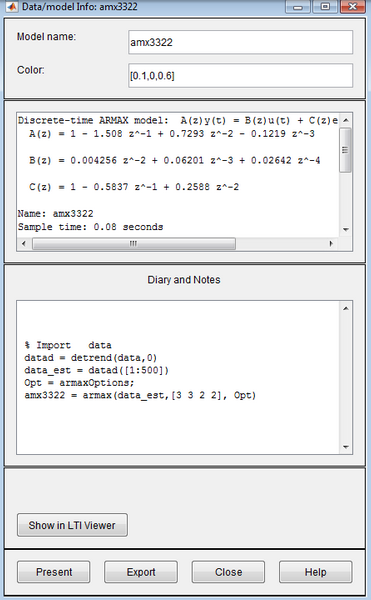


相比较之前的多了nc，nc就是噪声的阶数。

## 1.6：查看模型参数

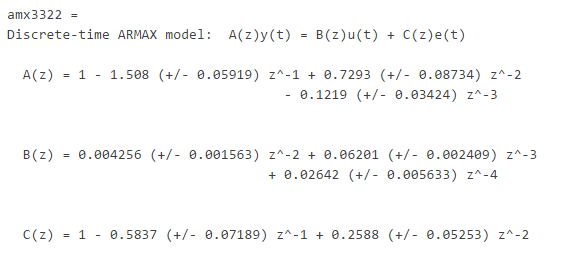
### 1.6.1：具体模型参数

在模型上面点击右键就会获取模型的具体参数。可见A,B,C都是具体的数据。



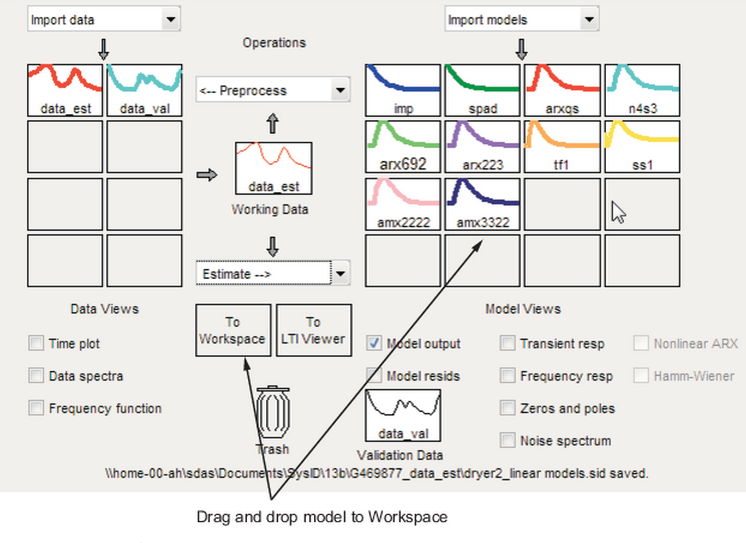
### 1.6.2：模型不确定性

点击上图中的present就可以在matlab中看到不确定的模型。都是具有一定的范围。



## 1.7：将模型导出到工作区

直接通过拖拽的方式。



## 1.8:导出到线性系统分析

可以直接将模型导出到LTI进行分析

# 2：matlab系统辨识工具箱命令行

命令行模式基本上和APP的流程是一致的。下面只介绍一些不同点

## 2.1：阶跃响应和频率响应

为什么要做这两种响应？因为它们可以直观的不想要参数的反映出系统的性能。例如延迟，不同频率的响应特性。

如果阶跃响应前面延迟大，那么对应的频率响应的相位就会下降的很快。

## 2.2：延迟估计

为什么要估计延迟？因为延迟是阶数的一部分。

对于SISO,可以从阶跃响应获取延迟

对于MISO，不同确定哪个输入导致了输出。可以通过写成微分方程的形式看出多久输出第一个数据。

## 2.3：阶数估计

为什么估计阶数？因为阶数和零极点和delay都有关系，并且对于黑盒子模型，必须输入阶数才能够进行估计。

多输入的系统，可以分开估计阶数，得到最好的匹配后，然后分别得到估计的模型。可得到分别的传递函数。

## 2.4：模型验证

比较匹配度。比较真实的输出和模型输出。

通过自相关和互相关来比较。Risdiual方式比较（还未搞清楚）

## 2.5：估计process模型

可以先假定零点个数，极点个数，积分，延迟等，建立一个模型，里面的参数都是未知的。并且可以给定各个参数的初始值和范围。

并且可以改变估计的算法。

## 2.6：估计黑盒子模型

用各种方式估计模型，和之前APP讲的各种模型都是一样的，只不过这里使用命令完成。

# 3：系统辨识APP识别低阶传递函数

## 3.1：简介

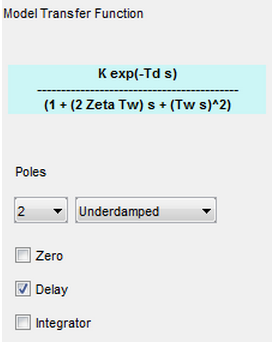
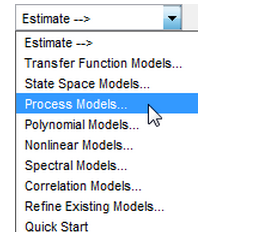
这里低阶是不高于三阶，通过改变零极点，积分和延迟，极点可以是实数或者是复数。例如

C:\Users\googol\AppData\Local\Temp\mx3137C.png

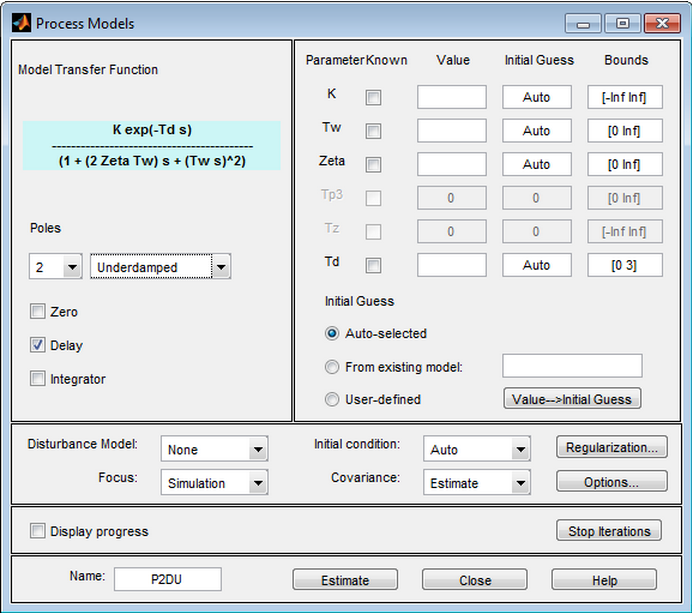
## 3.2：操作步骤（no -noise）

（1）：先做quick start 做数据的预处理

（2）：Estimate > Process models



该模块可以根据自己的先验信息，定义零极点，延迟等，并且可以给定初始值的范围。



如果你确定知道该参数，那么可以直接在initial guess中输入该值，选上known。假如知道参数的近似值，那么可以在initial guess中输入，并且将known单选框清除。

还可以选择focus，确定感兴趣的频率范围等。

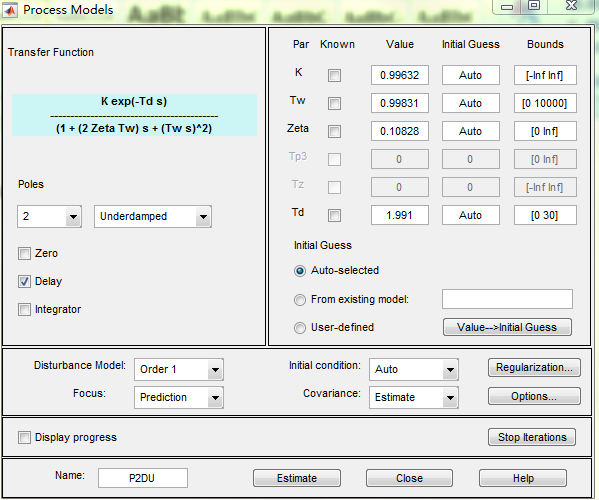
可以加扰动的模型等。

（3）：检测输出

检测模型的输出和实际测量的输出进行对比。通过相关性进行验证。（验证条件不明）

## 3.3：操作步骤（noise）

（1）数据处理省略，Estimate > Process models，然后在disturbance model选择一阶

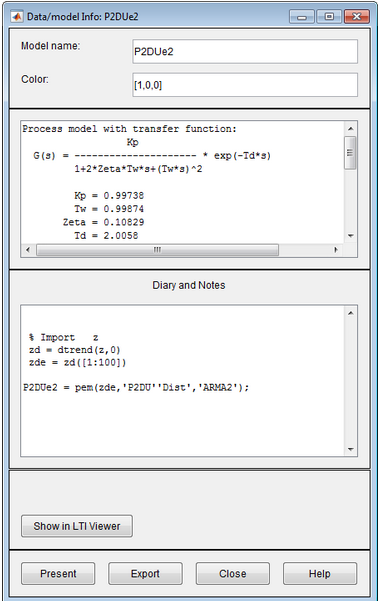


一阶噪声模型如下，其中D是一阶多项式，e是白噪声，这会使得focus改为prediction，可以提高频率域中噪声低部分的精度。

C:\Users\googol\AppData\Local\Temp\mx34308.png

（2）验证和上面方法一致。

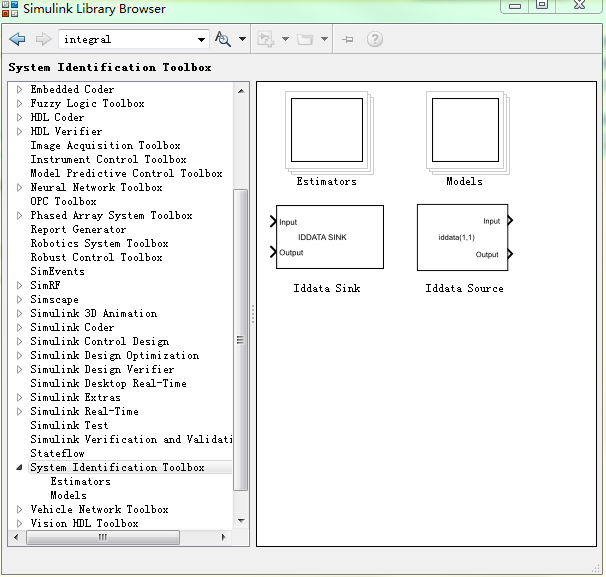
（3）查看模型参数，，其中还包含噪声的模型，还可显示参数的不确定性



## 3.4：在simulink中仿真系统辨识模型

（1）导入数据和模型在工作区中。其中模型是APP导出的模型。

（2）打开simulink，然后在系统辨识工具箱中，拖入输入，模型和输出

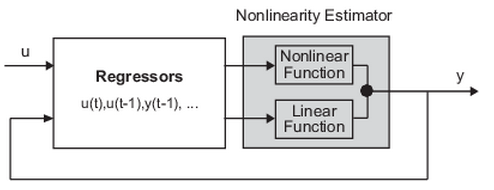


在输入中设置输入的数据，在model中导入之前在工作区的模型，还可以选择是否有噪声模型，在输出中设置输出的数据。即可。

# 4：APP识别非线性黑盒子模型

## 4.1:非线性模型类型

1：Nonlinear ARX models

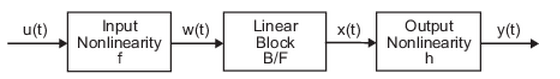


Regressors是回归量（当前的输入和过去的输入和过去的输出）。这个模块通过非线性和线性函数将regressors和输出联系起来。可以选择不同的非线性估计器。例如小波网络，神经网络等。

C:\Users\googol\AppData\Local\Temp\mx363C5.png

X是regressors矢量，L’(x-r)+d 是线性函数部分， *g*(*Q*(*x*−*r*))代表非线性部分输出，r是regressors X的均值。Q是映射矩阵。F(x)根据估计器不同而不同。

2：Hammerstein-Wiener models

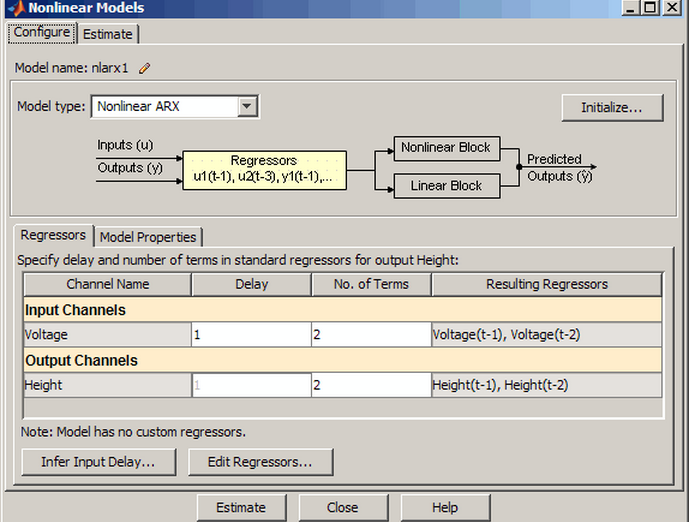
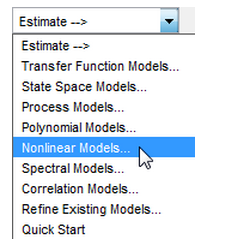


f和h是输入非线性和输出非线性。不同的输入需要有不同的f和h。中间是线性部分。W和x是中间变量。不需要同时都包含输入非线性和输出非线性。只有输入非线性的称为Hammerstein model。只有输出的称为Wiener model。

## 4.2：步骤

（1）：导入数据，分为估计数据和验证数据

（2）：非线性模型**Estimate** > **Nonlinear models**.



可以先设计regressors，delay 从1开始，有两个项。所以得到如下

C:\Users\googol\AppData\Local\Temp\mx36DD3.png

非线性的方式可以从model properties从选择。

（3）：估计验证。通过fit看是否匹配

（4）：对模型画出非线性的横截面。可看出其非线性特性。选择nolinear ARX即可显示该波形。

（5）：改变delay或者是model properties中的Number of units in nonlinear block .可以改变估计的模型。

（6）可从regressor中选择某一部分进行估计。 Edit Regressors

（7）可以基于之前的模型再选择不同的估计方法进行估计 Initialize

（8）选择最匹配的模型。

对于Hammerstein-Wiener models来说，需要设置输入输出Piecewise Linear（不知道是什么东东），还有线性部分的阶数和延迟。可以改变估计算法和延迟时间得到不同的模型。

