# 编程练习1:线性回归机器学习

简介：在练习1中，你将实现线性回归。在开始本练习之前，使用Octave中的cd命令切换到该目录。

本练习中包含的文件：

ex1.m - 用于练习单变量的Octave脚本

ex1 multi.m - 用于练习多变量的Octave脚本

ex1data1.txt - 一个变量的线性回归数据集

ex1data2.txt - 多个变量的线性回归数据集

**[1]** warmUpExercise.m - Octave中的简单示例函数

**[2]** plotData.m - 显示数据集的函数

**[3]** computeCost.m - 计算线性回归损失的函数

**[4]** gradientDescent.m –运行梯度下降函数

[5] computeCostMulti.m – 多变量成本函数

[6] gradientDescentMulti.m – 多变量梯度下降

[7] featureNormalize.m – 规范化特征函数

[8] normalEqn.m – 计算常规方程函数

**1-4：你必须要完成的部分**

**5-8：额外加分的部分**

在整个练习中，你将使用脚本ex1.m和ex1 multi.m。这些脚本为问题设置了数据集，并调用了将要编写的函数。你无需修改​​它们中的任何一个。你只需要按照此作业中的说明修改其他文件中的功能。

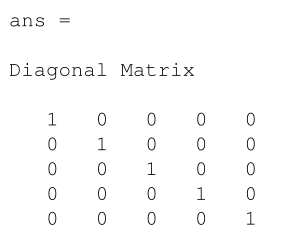
对于此编程练习，只需要完成练习的第一部分即可实现使用一个变量的线性回归。练习的第二部分，你可能会额外花些功夫完成，它涉及具有多个变量的线性回归。

# 1.简单的Matlab函数

ex1.m的第一部分为你提供Matlab语法和作业提交过程的练习。在文件warmUpExercise.m中，你将找到Matlab函数的轮廓。通过填写以下代码，对其进行修改以返回5 x 5的单位矩阵：

A = eye(5);

完成后，运行ex1.m（假设您位于正确的目录中，在Matlab提示符下键入“ ex1”），你将看到类似于以下内容的输出：



现在，ex1.m将暂停，直到你按任意键，然后再运行下一部分的代码。如果你希望退出，则键入ctrl-c将在程序运行中停止该程序

## 2.单变量线性回归

在本练习的这一部分中，您将使用一个变量实现线性回归，以预测食品卡车的利润。假设你是一家餐饮连锁店的首席执行官，并且正在考虑在不同的城市开设新的门店。该连锁店已经在各个城市开了卡车，您可以获得城市的利润和人口数据。

你想使用此数据来帮助你选择要扩展到下一个城市。

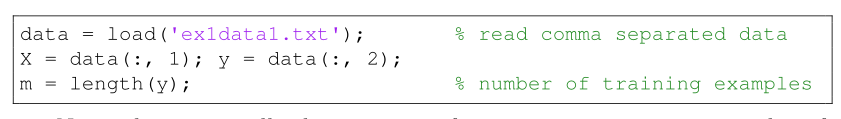
文件ex1data1.txt包含我们线性回归问题的数据集。第一列是城市的人口，第二列是该城市的餐车的利润。利润的负值表示亏损。

已经设置了ex1.m脚本来加载此数据。

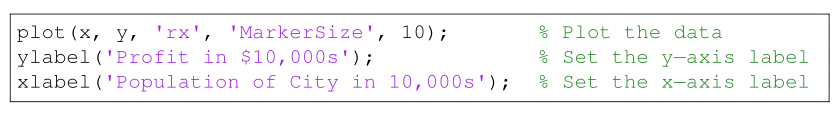
## 2.1绘制数据

在开始任何任务之前，通过可视化了解数据通常很有用。对于此数据集，你可以使用散点图来可视化数据，因为它仅具有两个要绘制的属性（利润和人口）。 （你在现实生活中会遇到的许多其他问题是多维的，无法在二维图上进行绘制。）

在ex1.m中，数据集从数据文件加载到变量X和y中：



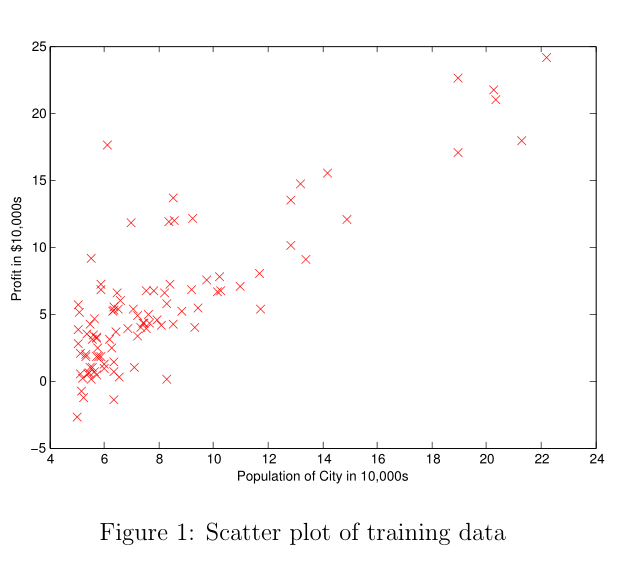
接下来，脚本调用plotData函数创建数据的散点图。你的工作是完成plotData.m绘制图；修改文件并填写以下代码：



现在，当你继续运行ex1.m时，最终结果应如图1所示，带有相同的红色“ x”标记和轴标签。

## 2.2梯度下降

在这一部分中，您将使用梯度下降法将线性回归参数θ拟合到我们的数据集中



## 2.2.1更新方程

线性回归的目的是使成本函数最小化





假设由线性模型给出：

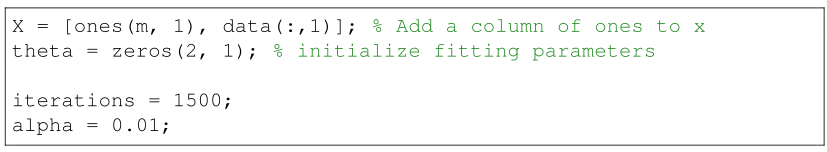


回想一下，模型的参数是。这些是您将调整以最小化成本的值。一种方法是使用批处理梯度下降算法。在批次梯度下降中，每次迭代都会执行更新(同时将所有的更新为)。

随着梯度的逐渐下降，参数将变得更接近于最低成本函数的。

实施注意：我们将每个示例作为一行存储在Matlab的X矩阵中。为了考虑截距项（），我们向X添加了第一列并将其设置为所有列。这使我们可以简单地将视为另一个“特征”。

## 2.2.2应用

在ex1.m中，我们已经设置了用于线性回归的数据。在接下来的几行中，我们向数据添加另一个维，以适应截距项。我们还将初始参数初始化为0，将学习率alpha初始化为0.01。

## 2.2.3计算成本J（θ）

当你执行梯度下降以学习最小化成本函数时，通过计算成本来监控收敛是有帮助的。在本节中，您将实现一个计算的函数，以便您可以检查梯度下降实现的收敛性。

你的下一个任务是完成文件computeCost.m中的代码，该文件是一个计算的函数。在执行此操作时，请记住，变量X和y不是标量值，而是其行代表训练集中的示例的矩阵

一旦完成该功能，ex1.m中的下一步将使用初始化为零的运行一次calculateCost，然后你将看到打印在屏幕上的成本。

你应该会看到成本为32.07。

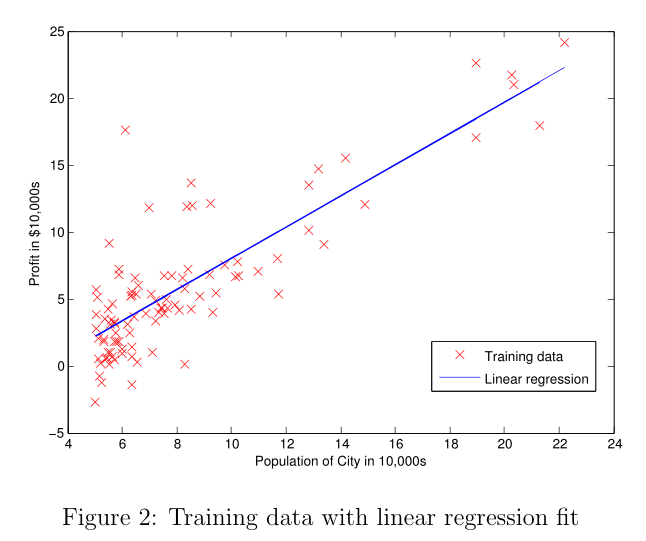
## 2.2.4 梯度下降

接下来，你将在gradientDescent.m文件中实现梯度下降。已经为您编写了循环结构，并且你只需要在每次迭代中为θ提供更新。

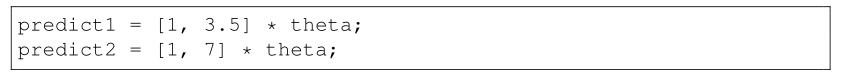
在编程时，请确保你了解要优化的内容和正在更新的内容。请记住，成本是由向量θ而不是X和y来参数化的。也就是说，我们通过改变矢量θ的值而不是通过改变X或y来最小化的值。

验证梯度下降是否正常工作的一种好方法是查看的值，并检查其每一步是否在减小。 gradientDescent.m的起始代码在每次迭代时都调用computeCost并打印成本。假设您正确实现了梯度下降和computeCost，则的值将永远不会增加，并且应在算法结束时收敛为稳定值。

完成后，ex1.m将使用最终参数来绘制线性拟合。结果应类似于图2：



你最终的θ值也将用于预测35,000和70,000人区域的利润。请注意，ex1.m中的以下各行使用矩阵乘法而不是显式求和或循环来计算预测的方式。这是Octave中代码矢量化的示例。



## 2.3调试

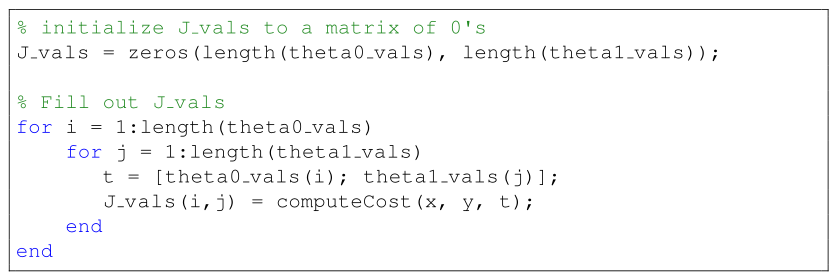
使用梯度下降时，请注意以下几点：

1. Matlab数组索引从1开始，而不是零。如果将θ0和θ1存储在称为theta的向量中，则值将为theta（1）和theta（2）。
2. 如果你在运行时发现许多错误，请检查矩阵操作，以确保要添加和乘以兼容尺寸的矩阵。使用size命令打印变量的尺寸将帮助您调试。
3. 默认情况下，Matlab将数学运算符解释为矩阵运算符。这是大小不兼容错误的常见原因。如果您不希望矩阵相乘，则需要添加“点”符号以将其指定为八度。例如，A \* B执行矩阵乘法，而A. \* B执行逐元素乘法

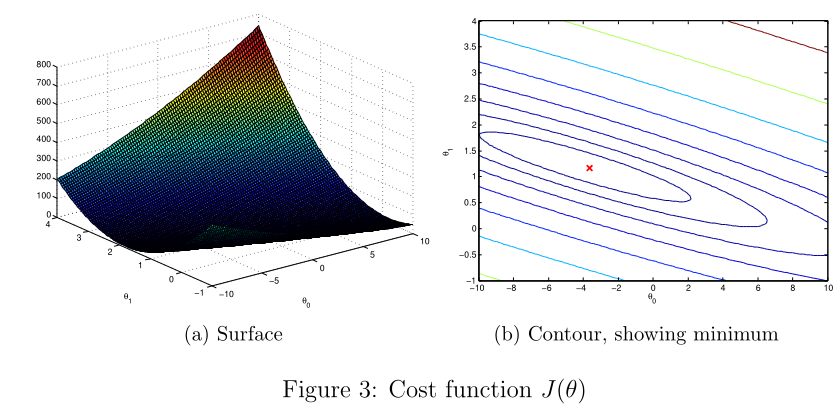
## 2.4可视化J(θ)

为了更好地理解成本函数，现在将在θ0和θ1值的二维网格上绘制成本。您无需为此部分编写任何新代码，但是你应该了解您已经编写的代码如何创建这些图像。

在ex1.m的下一步中，设置了代码，以使用你编写的computeCost函数在值的网格上计算。



执行完这些行后，您将拥有一个值的二维数组。然后，脚本ex1.m将使用这些值通过Surf和Contour命令生成的曲面和轮廓图。这些图应类似于图3：



这些图的目的是向您展示如何随θ0和θ1的变化而变化。成本函数是碗形的，并且具有全局最小值。 （这在轮廓图中比在3D表面图中更容易看到）。该最小值是θ0和θ1的最佳点，并且梯度下降的每一步都靠近该点。

## 3.附加加分练习（可选）

如果你已成功完成上述材料，那么恭喜！你现在已经了解了线性回归，并且应该能够在自己的数据集上开始使用它。

在本编程练习的其余部分，我们包括了以下可选的额外加分练习。这些练习将帮助你对材料进行更深入的了解，如果你能够做到，我们也鼓励你也完成它们。

## 4.多变量线性回归

在这一部分中，你将使用多个变量来实现线性回归，以预测房屋价格。假设你正在出售房屋，并且想知道一个好的市场价格。实现此目的的一种方法是，首先收集有关最近出售的房屋的信息，然后对房屋价格进行建模。

文件ex1data2.txt包含俄勒冈州波特兰市的房屋价格培训集。第一列是房屋的大小（以平方英尺为单位），第二列是卧室的数量，第三列是房屋的价格。

已经设置了ex1 multi.m脚本来帮助你逐步完成此练习。

## 4.1特征归一化

ex1 multi.m脚本将首先从该数据集中加载并显示一些值。通过查看这些值，请注意房屋大小约为卧室数量的1000倍。当特征相差一个数量级时，首先执行特征缩放可以使梯度下降收敛得更快。

你的任务是完成featureNormalize.m中的代码：

1. 从数据集中减去每个要素的平均值。
2. 减去平均值后，再将特征值除以它们各自的“标准差”。

标准差是一种测量特定特征的值的范围内有多少变化的方法(大多数数据点位于平均值的2个标准差内)；这是取值范围(最大-最小)的替代方法。在Octave中，你可以使用“标准”函数来计算标准偏差。例如，在featureNormalize.m中，数量X(:，1)包含训练集中x1(房屋大小)的所有值，因此std(X(:，1))计算房屋大小的标准偏差。在调用featureNormalize.m时，对应于x0= 1的1的额外列尚未添加到X(详见ex1 multi.m)。

你需要所有功能都这样做，并且你的代码应该可以处理所有大小的数据集(任何数量的功能/示例)。注意矩阵X的每一列对应一个特征。

**实现注意**:当标准化特征时，重要的是存储用于标准化的值——用于计算的平均值和标准差。在从模型中学习参数后，我们经常想预测我们以前没有见过的房子的价格。给定一个新的x值(客厅面积和卧室数量)，我们必须首先使用我们先前从训练集中计算的平均值和标准差来标准化x。

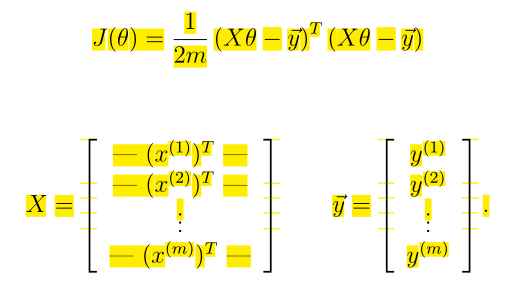
## 4.2梯度下降

在之前的实验中，你已经在一元回归问题上实现了梯度下降。现在唯一的区别是矩阵x中多了一个特征，假设函数和批量梯度下降更新规则保持不变。

你应该在computeCostMulti.m和gradientDescentMulti.m中完成代码，以实现多变量线性回归的成本函数和梯度下降。如果你在前面部分的代码(单个变量)已经支持多个变量，你也可以在这里使用它。

确保你的代码支持任意数量的特性，并且是良好矢量化的。您你可以使用“”来找出数据集中存在多少个要素。

实现注意:在多变量情况下，成本函数也可以用以下矢量化形式编写:



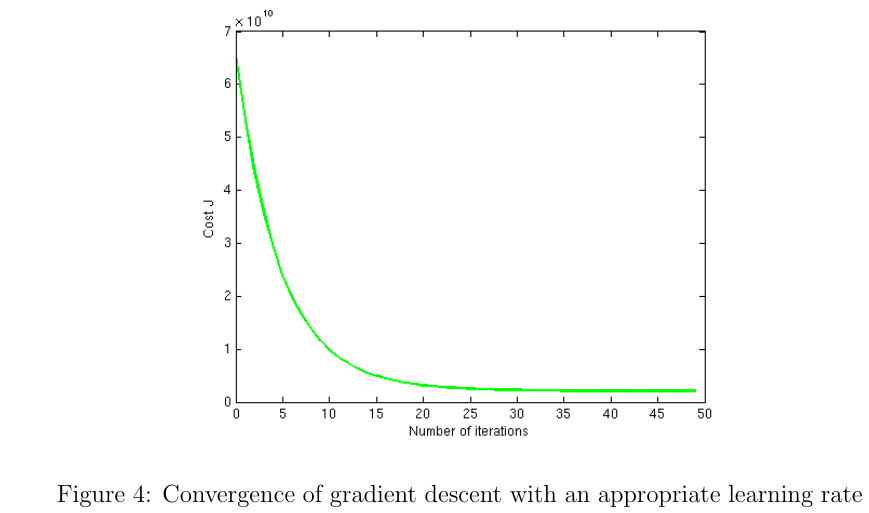
当你使用像Octave这样的数值计算工具时，矢量化版本是有效的。如果你是矩阵运算专家，你可以向自己证明这两种形式是等价的。

## 4.2.1 选择学习率

在练习的这一部分，你将尝试数据集的不同学习速率，并找到一个快速收敛的学习速率。你可以通过修改ex1 multi.m和更改设置学习速率的代码部分来更改学习速率。

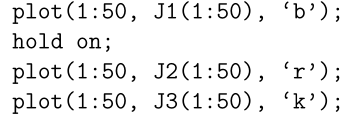
ex1 multi.m中的下一个阶段将调用你的gradientDescent.m函数，并以选定的学习速率运行梯度下降约50次迭代。该函数还应该返回向量J中J(θ)值的历史。在最后一次迭代之后，ex1 multi.m脚本根据迭代次数绘制J值。

如果你在一个好的范围内选择一个学习率，你的图类似于图4。如果你的图表看起来非常不同，特别是如果你的J(θ)值增加甚至爆炸，调整你的学习速度，再试一次。我们建议在对数标度上尝试学习率α的值，以大约3倍于先前值的乘法步长(即，0.3、0.1、0.03、0.01等等)。如果能帮助你看到曲线的整体趋势，你也可以调整你正在运行的迭代次数。



实施注意:如果你的学习率太大，J(θ)会发散和“爆炸”，导致数值对于计算机计算来说太大。在这些情况下，Matlab会倾向于返回NaNs。NaN代表“非数字”，通常由涉及∞和+∞的未定义操作引起。

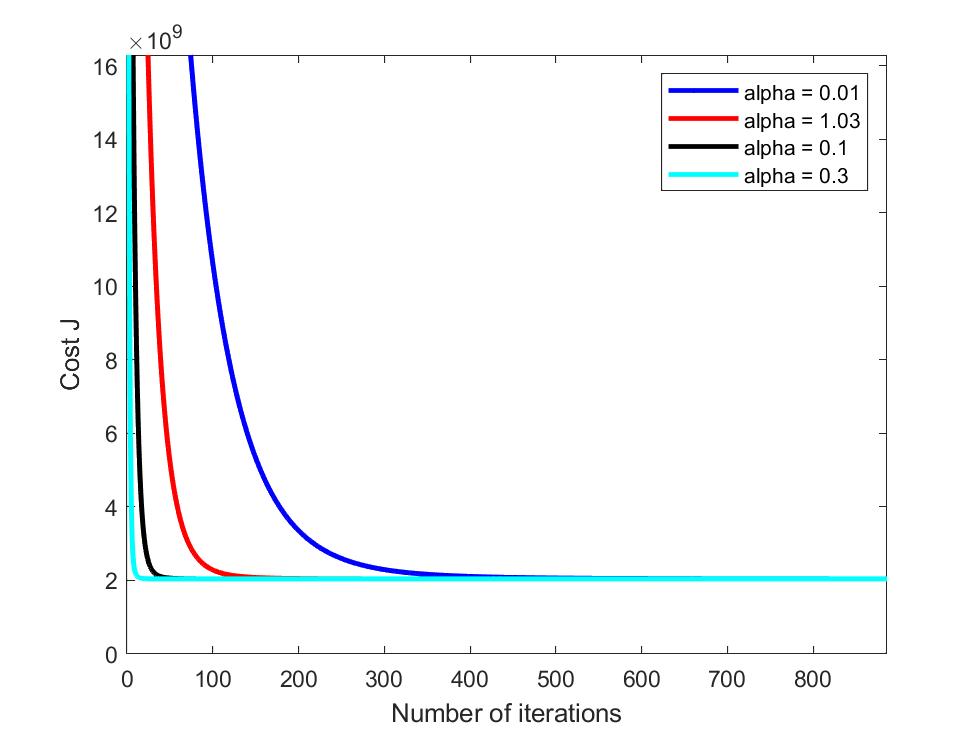
Matlab提示:为了比较不同的学习速率如何影响收敛，在同一图上绘制几个学习速率的J是有帮助的。在Matlab中，这可以通过在两个图之间使用“保持”命令多次执行梯度下降来完成。具体来说，如果您已经尝试了三个不同的alpha值(您可能应该尝试更多的值)，并将成本存储在J1、J2和JBOY3乐队，你可以使用以下命令将它们绘制在同一图上:



最后的参数“b”、“r”和“k”为图指定了不同的颜色。

注意，随着学习速率的变化，收敛曲线也会发生变化。学习率小的情况下，你应该会发现梯度下降需要很长时间才能收敛到最优值。相反，在学习率较大的情况下，梯度下降可能不会收敛，甚至可能会发散！

使用你找到的最佳学习速率，运行ex1 multi.m脚本运行梯度下降，直到收敛，以找到θ的最终值。接下来，用θ的这个值来预测一套1650平方英尺3卧室的房子的价格。稍后您将使用value来检查您对正规方程的实现。做这个预测的时候别忘了把特征归一化！



## 4.3标准方程

线性回归的封闭形式解是

使用这个公式不需要任何特征缩放，一次计算就可以得到精确解:不存在梯度下降中的“循环直到收敛”。

完成normalEqn.m中的代码，使用上面的公式计算θ。请记住，虽然你不需要缩放你的要素，但我们仍然需要在X矩阵中添加一列1，以获得截距项()。ex1.m中的代码会为您将1的列添加到X中。

练习:现在，一旦你用这种方法找到θ，就用它来预测一栋1650平方英尺、有3间卧室的房子的价格。你会发现，它给出的预测价格与你用梯度下降模型拟合得到的价格相同。

预测结果：

