成都大学
CHENGDU UNIVERSITY

《人工智能导论》
实验指导书

杨耀如

成都大学 电子信息与电气工程与工程学院

实验一　多变量函数作图及优化

一、实验目的

掌握对多变量函数作图和优化的方法；

掌握使用 MATLAB 产生多变量函数，并使用合适的作图函数对多变量函数进行刻画，并优化该多变量函数的方法。

二、实验内容

使用 MATLAB 编写代码，产生 Rosenbrock 函数，并按要求绘制和观察图形，并对该函数求解优化问题。

三、实验设备

1、计算机

2、MATLAB

四、实验原理

1、编写 Rosenbrock 函数的代码

Rosenbrock 函数被定义为

$$f(x,y)=100(y-x^2)^2+(x-1)^2$$

在定义域$x \in [-1.5,1.5], y \in [-0.5,2.1]$ 中，Rosenbrock 图像为下图的所示：

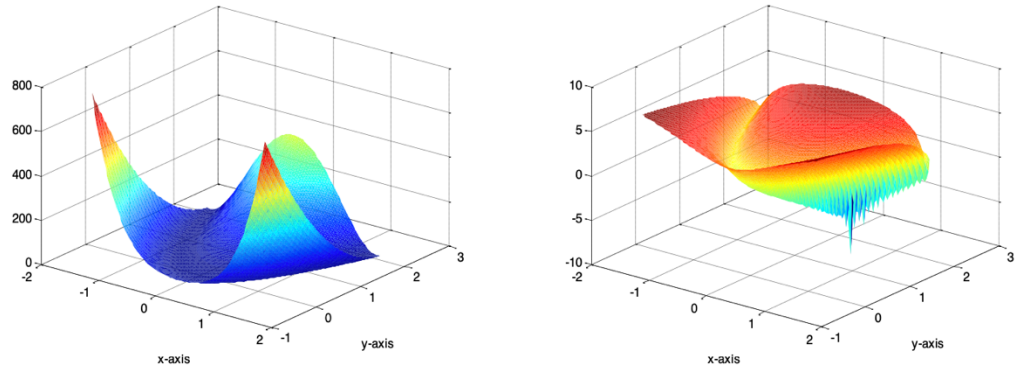

图 1（左）Rosenbrock 函数；（右）Rosenbrock 函数的对数在极值点附件展示了良好的结构

首先，编写 MATLAB 代码实现 Rosenbrock 函数，使用如下模版：

```
function z = rosenbrock(x, y)
```

在代码中使用“.”操作符使得函数可以将 2 维数组 x 和 y 作为自变量，对应的函数值同样应为一个 2 维数组。你的函数应该包含作图功能，能够做出和图 1 类似的函数图像。

使用 MATLAB editor，实现上面的 Rosenbrock 函数公式，并以 rosenbrock.m 的文件名保存。

2、调用 Rosenbrock 函数并作图

使用 MATLAB editor 新建一个脚本文件，输入下列代码，以 genrosen.m 的文件名保存。该代码实现的功能为调用你在 1 中所所实现的 Rosenbrock 函数并计算出正确的取值，并作图。

```
clear
% set resolution of sampling
nx = 100;
ny = 100;
% set bounds of space
x1 = -1.5;
x2 = 1.5;
y1 = -0.5;
y2 = 2.1;
% create 1-D parameters
X = linspace(x1,x2,nx);
Y = linspace(y1,y2,ny);
% transform 1-D parameters into 2-D form
[x, y] = meshgrid(X, Y);
% generate Rosenbrock space
z = rosenbrock(x,y);
% do a default, pretty surface plot
surf(x,y,z)
xlabel('x-axis')
ylabel('y-axis')
shading interp
view(37.5,30)
```

在你的 genrosen.m 代码中合适的位置加入 title()函数来打印你自己的姓名和学号，否则视为未完成。

使用 MATLAB editor 新建一个脚本文件，输入下列代码，以 contourrosen.m 的文件名保存。该代码实现的功能为使用 max(),min()和 logspace()函数在定义域中函数的极大值点和极小值点之间的对数空间中将函数的对数值 20 等分，并作出 contour 图。你应该在运行 contourrosen.m 后看到一个和图 2 相似的图像。我们会在后续实验中使用该图像来可视化优化的过程。

```
% must be run after the "genrosen.m" file
% get vertical bounds of space and generate countours
maximum = max(max(z));
minimum = min(min(z));
contours=logspace(log10(minimum),log10(maximum),20);
contour(x,y,z,contours)
```

在你的 contourrosen.m 代码中合适的位置加入 title()函数来打印你自己的姓名和学号，否则视为未完成。

3、Rosenbrock 函数的梯度

在后续实验中我们使用跟梯度相关的方法来对 Rosenbrock 函数进行优化。因此，实现一个能计算 Rosenbrock 函数在定义域内任意点的梯度的函数显得很有必要。尽管

Rosenbrock 函数的梯度的解析式很容易求得，但是这里，我们需要通过数值而不是解析的方法来求解梯度。

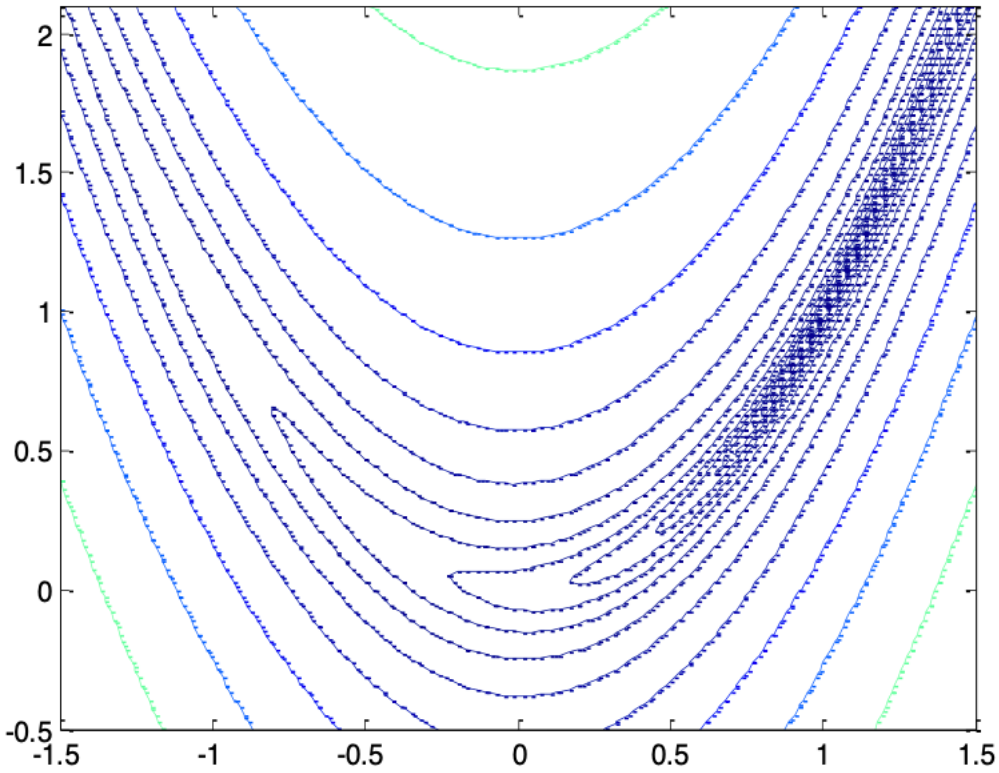

图 2Rosenbrock 函数的对数值的 contour 图像

对于单变量函数$f(x)$，其在点x_0时，有：

$$f'(x_0) \underset{h\to 0}{=} \frac{f(x_0+h)-f(x_0-h)}{2h}$$

将之推广到 2 维的情况，函数$f(x,y)$，在点(x_0,y_0)时，其梯度为：

$$\nabla f(x_0,y_0) \underset{h\to 0}{=} \frac{f(x_0+h,y_0)-f(x_0-h,y_0)}{2h}\hat{\mathbf{x}} + \frac{f(x_0,y_0+h)-f(x_0,y_0-h)}{2h}\hat{\mathbf{y}}$$

因此，选择合适的参数h,我们可以根据上式来实现数值方法近似地计算梯度。

使用 MATLAB editor，根据上面的数值计算梯度原理，实现计算 Rosenbrock 函数在任意点的梯度的两个分量的函数函数，并以 gradrosen.m 的文件名保存。该函数模版如下：

```
function g = gradrosen(x1,y1)

... function body ...
```

该函数应返回向量 g=[gx, gy]，gx 和 gy 表示梯度沿 X 和 Y 方向的分量：

$$gx = \frac{f(x_0+h,y_0)-f(x_0-h,y_0)}{2h}$$

$$gy = \frac{f(x_0,y_0+h)-f(x_0,y_0-h)}{2h}$$

对于参数h的取值，直接影响到梯度计算的健壮性。一个可行的策略是一个初始的h值

开始，计算梯度值，然后减小h的值，再重新计算梯度值，我们可以通过测试梯度值改变的大小来检验h的取值是否合适：如果两次计算的梯度值改变很小，则是可以接受的；若两次计算的梯度值改变较大，则需要再次减小h的值，再重新计算梯度进行比较。因此，以下的伪代码应是你的 gradrosen.m 应体现的思路。

```
Calculate initial value of gradient
While error in gradient is too high
      Calculate new value of h
      Calculate new value of gradient
      Calculate error between gradient estimates
End while
Return gradient
```

一个可以参考的更新h值的策略是除以 2 或 10。请思考初始化时合理的h初始值有什么特点，为什么将h初始化为很小的值是危险的。

对于每次比较梯度值的改变时，绝对误差(absolute error)或相对误差(relative error)都是可以考虑的，请思考为什么相对误差是一个更健壮的指标？对于同一个值的两次估计，若记为x_{old}和x_{new}，其相对误差定义为：

$$error = \left| \frac{x_{new} - x_{old}}{x_{new}} \right|$$

在代码实现时，还有一个实际问题需要考虑：当梯度值极小时，会发生什么？如何在代码中实现附加的逻辑来测试这一情况是否出现并避免这一情况的出现(即 gx 和/或 gy 极小时)。

4、最速下降法（梯度下降法）优化

最简单的最速下降法只需要很简短的代码就能实现，其实现的伪代码如下：

```
While we wish to continue
      Calculate gradient at a point
      Take a step based on this gradient
End while
```

如果从点(x_0, y_0)开始，为了更新到下一点，我们使用梯度$\nabla f(x_0, y_0)$和步长d，因此，最速下降（梯度下降法）第一次更新可以写为：

$$(x_1, y_1) = (x_0, y_o) - \nabla f(x_0, y_0)\, d$$

若写为两个坐标方向，即为：

$$x_1 = x_0 - g_x\, d$$

$$y_1 = y_0 - g_y\, d$$

按照上述数学原理，使用 MATLAB 编写一个脚本，实现使用最速下降（梯度下降）法来优化 Rosenbrock 函数，建议先选择起始点$(-1, 0)$，使用固定的步长参数$d = 0.001$，更新轮数 100。

在图 2 上做出最速下降的轨迹（提示：使用 plot 函数在图 2 的 contour 图上画出梯度下降的轨迹，查询“hold on”和“hold off”命令的帮助文档，思考如何实现在 contour 图上叠加地画出更新轨迹。）。调整步长和更新轮次的参数值，观察这些参数对收敛性的影响，至少尝试 3 组不同的步长和更新轮次的参数组合，每组都要画图。将该脚本以 graddcnt.m 的文件名保存。

在你的 graddcnt.m 代码中合适的位置加入 title()函数来打印你自己的姓名和学号，否则视为未完成。

5、提升收敛性

一系列提升收敛性的方法可以用来提升最速下降法的收敛性。一个最简单思路是从起始点开始，一直使用起始点的梯度值来进行梯度下降更新，考察每次更新后 Rosenbrock 函数值，直到当一次更新后函数值开始增大，则将该点视为起始点，重新计算梯度值，执行梯度下降。该思路的伪代码如下：

```
While we wish to continue
      Calculate gradient at a point
      Take a step based on this gradient
      Evaluate the function at the new point
      While the new value is less than the old value
            Take another step based on current estimate of gradient
            Evaluate the function at the new point
      End while
End while
```

编写 MATLAB 代码实现最速下降法的这一改进，并在图 2 的 contour 图上做出改进后的最速下降法更新轨迹，将相关代码以 graddcntimprv.m 的文件名保存。

在你的 graddcntimprv.m 代码中合适的位置加入 title()函数来打印你自己的姓名和学号，否则视为未完成。

6、进一步收敛性

首先，引入一个收敛参数，通常记为ε。为了使得程序能够在任意更新轮次后自动终止循环（通常情况下，可用“while”循环代替“for”循环），对于下降算法，MATLAB 中一个通常的收敛性测试是$norm(\nabla f) < \varepsilon$，这是一个对梯度大小的测试指标。

然后，引入一个可变的步长。在本例子中，可将$norm(\nabla f)$与步长做比较，当这两个量的模在数量级上比较接近时，可减小步长参数大小。在本例中，可采取“当$norm(\nabla f) < step * 5$时，则将步长参数减半”的策略。

编写 MATLAB 代码实现最速下降法的这一改进，并在图 2 的 contour 图上做出进一步改进后的最速下降法更新轨迹，将相关代码以 graddcntfurtherimprv.m 的文件名保存。

在你的 graddcntfurtherimprv.m 代码中合适的位置加入 title()函数来打印你自己的姓名和学号，否则视为未完成。

7、再再再提升收敛性：共轭梯度优化法（选做）

使用梯度$\boldsymbol{g} = \nabla f(x, y)$，来计算一个共轭方向，d，我们希望更新沿着 d 方向进行。伪代码如下：

Calculate $\mathbf{g}_0 = \nabla f(x_0, y_0)$; $\mathbf{d}_0 = -\mathbf{g}_0$

While we wish to continue

Calculate new point using $(x_1, y_1) = (x_0, y_0) + step * \mathbf{d}_0$

Calculate gradient at this point, $\mathbf{g}_1$

Calculate new conjugate direction, $\mathbf{d}_1$

Set $\mathbf{d}_0 = \mathbf{d}_1$, etc

End while

其中最核心的部分是共轭方向的更新，这里，我们采用策略：

$$\mathbf{d}_1 = -\mathbf{g}_1 + \beta \mathbf{d}_0$$

其中的参数β，可通过如下公式计算：

$$\beta = \frac{\mathbf{g}_1^T \mathbf{g}_1}{\mathbf{g}_0^T \mathbf{g}_0}$$

编写 MATLAB 代码实现共轭梯度优化法，并在图 2 的 contour 图上做出进一步改进后的最速下降法更新轨迹，将相关代码以 conjgraddcnt.m 的文件名保存。建议先选择起始点$(-1, 0)$，使用固定的步长参数$d = 0.001$，更新轮数 100。

8、再再再再提升收敛性（选做）

按照 5 中对最速下降法的改进，对共轭梯度优化法进行改进。

编写 MATLAB 代码实现改进后的共轭梯度优化法，将相关代码以 conjgraddcntimprv.m 的文件名保存。

9、再再再再再提升收敛性（选做）

按照 6 中对最速下降法的改进，对共轭梯度优化法进行进一步改进。

编写 MATLAB 代码实现改进后的共轭梯度优化法，将相关代码以 conjgraddcntfurtherimprv.m 的文件名保存。

五、实验要求

1、编写 MATLAB 代码，至少实现四.1、2、3、4、5、6 的要求；

2、绘制典相关图形的图形；

3、分析实验结果（回答四.2 和四.3 中的问题）。