日期： 2023 年 5 月 18 日

中山大學 SUN YAT-SEN UNIVERSITY

成绩： _____

学院：智能工程学院　　课程：最优化理论与方法　　周次：第 13 周

专业：智能科学与技术　　姓名：方桂安　　学号：20354027

1 题一

1.1 题目

利用 MATLAB 编程实现: 采用最速梯度下降法求如下函数的极小点, 初始值为

$$\mathbf{x}(0) = \begin{bmatrix} 2 & 1 \end{bmatrix}^T, \quad \varepsilon = 10^{-2}$$

$$f(x) = \frac{1}{2}x_1^2 + x_2^2$$

1.2 解答

算法分析：最速梯度下降法是一种迭代优化算法，用于求解无约束优化问题。在每次迭代中，算法沿着负梯度方向搜索，直到找到一个满足收敛条件的解。

函数的梯度为：

$$\nabla f(x) = \begin{bmatrix} x_1 \\ 2x_2 \end{bmatrix}$$

在每次迭代中，我们需要找到一个合适的步长 α，使得沿着负梯度方向的搜索能够使函数值减小。我们可以使用一维搜索方法（如黄金分割法或者斐波那契搜索法）来寻找最优的步长。

编程解答：

```
% 利用最速梯度下降法求解函数的极小点

% 函数定义
f = @(x) 0.5*x(1)^2 + x(2)^2;

```

```
% 梯度定义
grad_f = @(x) [x(1); 2*x(2)];

% 初始值和收敛阈值
x0 = [2; 1];
epsilon = 1e-2;

% 最速梯度下降法
alpha = 0.1; % 初始步长
x = x0;
grad = grad_f(x);
iter = 0;

while norm(grad) > epsilon
    % 计算步长
    % 这里可以使用线搜索方法来选择合适的步长 alpha，如 Armijo 规则或 Wolfe 条件

    % 更新变量
    x = x - alpha * grad;
    grad = grad_f(x);

    iter = iter + 1;
end

% 输出结果
fprintf('最小值点: (%f, %f)\n', x(1), x(2));
fprintf('迭代次数: %d\n', iter);
```

最终结果为：最小值点: (0.009277, 0.000011)；迭代次数: 51。

2 题二

2.1 题目

利用 MATLAB 编程实现: 采用牛顿法求如下函数的极小点, 初始值为 $\mathbf{x}(0)=\begin{bmatrix} -2 & 2 \end{bmatrix}^T$。当函数的梯度范数小于 10^{-2} 时, 停止迭代。

$$f(x)=100\left(x_2-x_1^2\right)^2+\left(1-x_1\right)^2$$

2.2 解答

算法分析：牛顿法是一种迭代优化算法，用于求解无约束优化问题。在每次迭代中，算法使用目标函数的二阶导数信息（Hessian 矩阵）来更新解。牛顿法的收敛速度通常比梯度下降法更快。

函数的梯度为:

$$\nabla f(x) = \begin{bmatrix} -400x_1\left(x_2 - x_1^2\right) - 2\left(1 - x_1\right) \\ 200\left(x_2 - x_1^2\right) \end{bmatrix}$$

函数的 Hessian 矩阵为:

$$H(x) = \begin{bmatrix} 1200x_1^2 - 400x_2 + 2 & -400x_1 \\ -400x_1 & 200 \end{bmatrix}$$

在每次迭代中，我们使用以下更新公式:

$$x^{(k+1)} = x^{(k)} - H\left(x^{(k)}\right)^{-1} \nabla f\left(x^{(k)}\right)$$

编程解答:

```
% 利用牛顿法求解函数的极小点

% 函数定义
f = @(x) 100*(x(2) - x(1)^2)^2 + (1 - x(1))^2;

% 梯度定义
grad_f = @(x) [-400*x(1)*(x(2) - x(1)^2) - 2*(1 - x(1));
               200*(x(2) - x(1)^2)];

% Hessian 矩阵定义
hessian = @(x) [1200*x(1)^2 - 400*x(2) + 2, -400*x(1);
                -400*x(1), 200];

% 初始值和收敛阈值
x0 = [-2; 2];
epsilon = 1e-2;

% 牛顿法
x = x0;
grad = grad_f(x);
iter = 0;

while norm(grad) > epsilon
    % 计算 Hessian 矩阵和其逆矩阵
    H = hessian(x);
    inv_H = inv(H);

    % 更新变量
    x = x - inv_H * grad;
    grad = grad_f(x);
```

```
31
32      iter = iter + 1;
33  end
34
35  % 输出结果
36  fprintf('最小值点: (%f, %f)\n', x(1), x(2));
37  fprintf('迭代次数: %d\n', iter);
```

最终结果为：最小值点: (1.000000, 1.000000); 迭代次数: 5。

3 题三

3.1 题目

利用 MATLAB 编程实现: 共轭梯度法求如下方程组的根, 初始值为 $\mathbf{x}(0) = \begin{bmatrix} 0 & 0 \end{bmatrix}^T$。

$$\begin{cases} 4x_1 - 3x_2 = 11 \\ 2x_1 + x_2 = 13 \end{cases}$$

3.2 解答

算法分析：共轭梯度法是一种迭代优化算法，用于求解线性方程组 $Ax = b$，其中 A 是对称正定矩阵。共轭梯度法的优点是收敛速度较快，尤其适用于大规模稀疏矩阵的求解。

在每次迭代中，共轭梯度法使用当前残差和之前的搜索方向来更新解。搜索方向在每次迭代中都被更新，以保持与之前的搜索方向共轭。

对于共轭梯度法，我们需要确保系数矩阵 A 是对称正定的。在这个问题中，系数矩阵为：

$$A = \begin{bmatrix} 4 & -3 \\ 2 & 1 \end{bmatrix}$$

通过以下代码判断，可以得出结论矩阵 A 不符合条件。

```
1  % 定义矩阵
2  A = [4, -3;
3       2,  1];
4
5  % 判断对称性
6  if isequal(A, A')
7      % 计算特征值
8      eigenvalues = eig(A);
9
```

```
    % 判断正定性
    if all(eigenvalues > 0)
        disp('矩阵是对称正定的');
    else
        disp('矩阵不是对称正定的');
    end
else
    disp('矩阵不是对称矩阵');
end
```

由于系数矩阵 A 不是对称正定的，共轭梯度法可能无法收敛到正确的解。可以尝试以下方法解决问题：

- 预条件共轭梯度法（PCG）：预条件共轭梯度法是共轭梯度法的一种改进，通过引入一个预条件矩阵 M 来改善矩阵 A 的性质。预条件矩阵 M 应该是正定的，并且接近 A 的逆矩阵。这样，新的线性方程组 $M^{-1}Ax = M^{-1}b$ 的系数矩阵 $M^{-1}A$ 将具有更好的性质，从而使共轭梯度法能够收敛到正确的解。常用的预条件矩阵包括不完全 Cholesky 分解、不完全 LU 分解等。

- 将 A 转化为对称正定矩阵：如果可能，可以尝试将原始方程组的系数矩阵 A 转化为对称正定矩阵。例如，如果 A 是非奇异的，可以构造新的线性方程组 $A^TAx = A^Tb$。新的系数矩阵 A^TA 是对称正定的，因此可以使用共轭梯度法求解。

需要注意的是，这些方法可能会增加计算复杂度或引入额外的误差。这里其实考虑使用其他线性方程组求解方法，如高斯消元法、LU 分解法或迭代法（如雅可比迭代法或高斯-赛德尔迭代法）等，也可以轻松地解出根来。本次作业中我采取的是将 A 转化为对称正定矩阵。

编程解答：

```
% 利用共轭梯度法求解方程组的根

% 原始方程组的系数矩阵 A
A = [4, -3; 2,  1];

% 原始方程组的右侧向量 b
b = [11; 13];

% 转化为对称正定矩阵 B = A^T * A
B = A' * A;

% 转化后的方程组的右侧向量
b_new = A' * b;

% 初始值和收敛阈值
x0 = [0; 0];
```

```
epsilon = 1e-6;

% 共轭梯度法
x = x0;
r = b_new - B * x;
p = r;
iter = 0;

while norm(r) > epsilon
    alpha = (r' * r) / (p' * B * p);
    x = x + alpha * p;
    r_new = r - alpha * B * p;
    beta = (r_new' * r_new) / (r' * r);
    p = r_new + beta * p;
    r = r_new;
    iter = iter + 1;
end

% 输出结果
fprintf('方程的根: (%f, %f)\n', x(1), x(2));
fprintf('迭代次数: %d\n', iter);
```

最终结果为：方程的根: (5.000000, 3.000000)；迭代次数: 2。

4 题四

4.1 题目

利用 MATLAB 编程实现：**DFP** 算法求如下函数的极小点。令起始点为

$$\mathbf{x}(0) = \begin{bmatrix} 1 & 1 \end{bmatrix}^T, H_0 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$f(x) = x_1^2 + 2x_2^2 - 4x_1 - 2x_1x_2$$

4.2 解答

算法分析：DFP（Davidon-Fletcher-Powell）算法是一种拟牛顿法，用于求解无约束优化问题。拟牛顿法的核心思想是在每次迭代中使用一个近似的 Hessian 矩阵来替代真实的 Hessian 矩阵。DFP 算法通过使用梯度信息来更新近似的 Hessian 矩阵，从而避免了计算真实 Hessian 矩阵的高计算成本。

函数的梯度为：

$$\nabla f(x) = \begin{bmatrix} x_1 \\ 2x_2 \end{bmatrix}$$

在每次迭代中，我们需要找到一个合适的步长 α，使得沿着负梯度方向的搜索能够使函数值减小。我们可以使用固定步长或者自适应步长策略。

编程解答：

```
% 利用 DFP 算法求解函数的极小点

% 函数定义
f = @(x) x(1)^2 + 2*x(2)^2 - 4*x(1) - 2*x(1)*x(2);

% 梯度定义
grad_f = @(x) [2*x(1) - 4 - 2*x(2); 4*x(2) - 2*x(1)];

% 初始点和初始近似 Hessian 矩阵
x0 = [1; 1];
H0 = eye(2);

% 最大迭代次数和停止迭代的阈值
max_iter = 100;
epsilon = 1e-6;

% DFP 算法
x = x0;
H = H0;
g = grad_f(x);
iter = 0;

while norm(g) > epsilon && iter < max_iter
    d = -H * g; % 计算搜索方向

    % 使用线搜索方法选择合适的步长
    alpha = 1; % 这里可以使用固定步长或者其他线搜索方法

    x_new = x + alpha * d;
    g_new = grad_f(x_new);
    s = x_new - x;
    y = g_new - g;

    rho = 1 / (y' * s);
    H = (eye(2) - rho * s * y') * H * (eye(2) - rho * y * s') + rho * s * s'; % 更新近似 Hessian 矩阵

    x = x_new;
    g = g_new;
    iter = iter + 1;
end

% 输出结果
```

```
43  fprintf('最小值点: (%f, %f)\n', x(1), x(2));
44  fprintf('迭代次数: %d\n', iter);
```

最终结果为：最小值点: (4.000000, 2.000000)；迭代次数: 2。

5 题五

5.1 题目

前半学期课程总结，包括：

1. 知识点总结与分析；
2. 个人角度的课程难点；
3. 前半学期的最优化方法在自己其他课程和科研工作中的应用设想。

5.2 知识点总结与分析

首先对于知识点总结与分析，我采取思维导图的方式展示，将我平常在课上所做的 markdown 笔记利用 markmap 转换成 pdf，展示如图1。当然由于篇幅限制，这里的观看效果较差，老师可以通过我的博客来查阅。

在前半学期里，我们学习了最优化的简介、方法概述、无约束优化最优化条件、线搜索方法、信赖域方法、最速下降算法与牛顿算法、共轭梯度法、拟牛顿法以及最小二乘问题。下面是对这些知识点的总结与分析：

最优化简介：最优化是应用数学的一个分支，主要研究在特定情况下最大化或最小化某一特定函数或变量。最优化问题通常需要对实际需求进行定性和定量分析，建立恰当的数学模型来描述该问题，设计合适的计算方法来寻找问题的最优解，探索研究模型和算法的理论性质，考察算法的计算性能等多方面。

无约束优化最优化条件：一阶最优性条件是指，如果一个多元函数在某点取得局部最大值或最小值，那么该点的梯度为零。二阶充分最优性条件是指，如果一个二次连续可微函数在某个驻点处的海森矩阵是正定的（或负定的），那么该点是局部最小值点（或局部最大值点）。

无约束优化线搜索方法：线搜索方法是一种求解无约束优化问题的方法，主要关注步长和方向的选择。步长的选择引入了重要概念 line search，方向的选择引入了重要概念 Quasi-Newton Method。

信赖域方法：信赖域方法是一种求解非线性优化问题的方法，其基本思想是利用一个局部模型来近似当前迭代点附近的函数情况。信赖域子问题的目的是寻找近似函数在信赖域内取最小时对应的自变量取值。

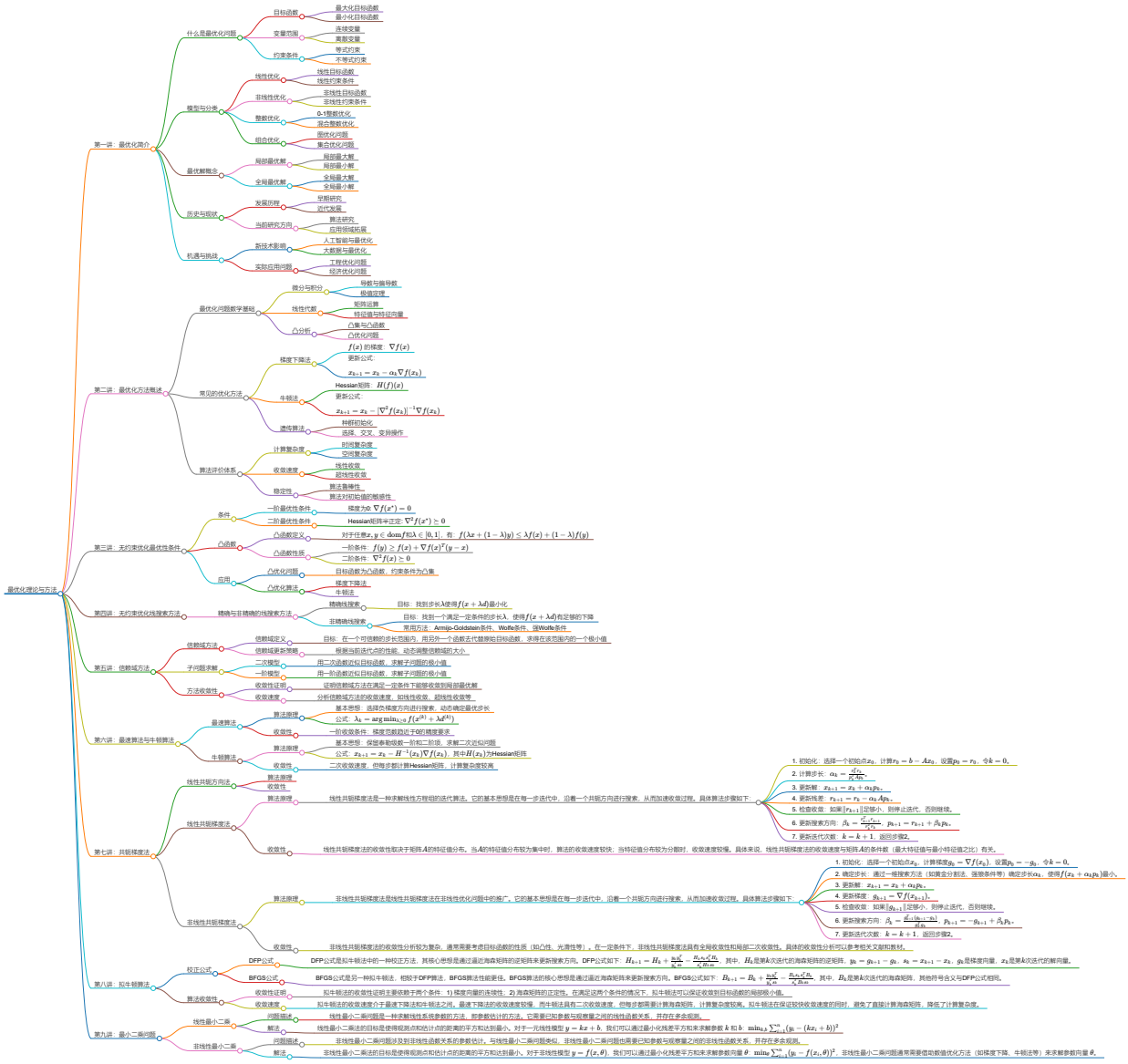

图 1: 最优化期中知识思维导图

最速下降算法与牛顿算法：最速下降算法是一种基于梯度的优化方法，每次迭代都沿着负梯度方向进行。牛顿算法则是一种基于二阶导数信息的优化方法，每次迭代都沿着海森矩阵的逆乘以负梯度方向进行。

共轭梯度法：共轭梯度法是一种求解线性方程组的迭代方法，特别适用于求解大规模稀疏线性方程组。它的优点是收敛速度快，且不需要计算海森矩阵的逆。

拟牛顿法：拟牛顿法是一种求解无约束优化问题的方法，它试图在牛顿法的基础上减少计算量。拟

牛顿法的核心思想是用一个矩阵来近似海森矩阵的逆，从而避免了直接计算海森矩阵的逆。

最小二乘问题：最小二乘问题是一种求解线性或非线性方程组的方法，主要关注如何最小化误差函数的平方和。常见的最小二乘问题求解方法有牛顿法、梯度下降法、高斯牛顿法和列文伯格-马夸特法。

5.3 个人角度的课程难点

1. 理解最优化问题的数学模型：最优化问题的数学模型涉及到复杂的函数和约束条件，理解这些模型需要较强的数学基础，在推导这部分内容的时候我感到比较吃力，例如 KKT 条件。

2. 选择合适的优化算法：针对不同的问题，需要选择合适的优化算法。这需要对各种算法的性能和适用范围有深入的了解，目前我对这方面经验比较薄弱，难以作出最好的选择。

3. 实际应用中的调参和优化：在实际应用中，优化算法的性能可能受到参数设置、初始值选择等因素的影响。如何调整这些参数以获得更好的优化效果是一个具有挑战性的问题。需要较多的工程经验，目前我在解决问题时还处于随机取值的阶段。

5.4 最优化方法的应用设想

5.4.1 在其他课程中的应用

最优化理论与算法在机器学习、深度学习和计算机视觉中有广泛的应用。以下是一些典型的应用场景：

- 机器学习：几乎所有的机器学习算法都可以归结为求一个目标函数的极值，例如有监督学习中的损失函数最小化问题。最优化算法在机器学习中的应用包括参数估计、模型选择和特征选择等。

- 深度学习：深度学习中的神经网络训练过程本质上是一个最优化问题。优化算法用于更新神经网络参数，使损失函数最小化。常见的优化算法包括梯度下降法、随机梯度下降法、小批量梯度下降法、动量法、AdaGrad、RMSProp、Adam 等。

- 计算机视觉：计算机视觉中的许多任务，如目标识别、目标跟踪、视频内容理解、图像去噪、三维重建和目标三维姿态估计等，都可以通过最优化方法来求解。例如，在图像去噪问题中，可以构造一个合适的目标函数，使得该目标函数取到极值的解就是去噪后的图像。

总之，最优化理论与算法在机器学习、深度学习和计算机视觉中发挥着关键作用。它们为这些领域提供了一种通用的求解框架，帮助研究人员和工程师解决各种实际问题。

5.4.2 个人科研工作

目前我所研究的课题是利用扩散模型进行 2d/3d 的生成，接下来我简单介绍一下。

扩散模型（Diffusion Model）是一种基于去噪技术的图像生成方法。其基本思想是让计算机自动学习一些数据的统计规律，并利用这些规律生成新的数据，如图像、音频等。扩散模型的生成过程包括两个步骤：

- 前向扩散过程：逐步向图片增加噪声，直到最终得到一张纯噪声图片。在这个过程中，我们有：

$$x_t = \sqrt{\alpha_t} x_{t-1} + \sqrt{1-\alpha_t}\epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, I)$$

  其中，x_t 表示在时间步 t 的图像，α_t 是一个预先定义的系数，ϵ_t 是从标准正态分布采样的噪声。

- 反向去噪过程：训练一个神经网络逐渐地从一张纯噪声图片中消除噪声，直到得到一张真实的图片。在这个过程中，我们有：

$$x_{t-1} = \mu_\theta\left(x_t, t\right) + \sigma_\theta\left(x_t, t\right)\epsilon_t$$

  其中，$\mu_\theta(x_t, t)$ 和 $\sigma_\theta(x_t, t)$ 是神经网络预测的均值和方差，ϵ_t 是从标准正态分布采样的噪声。

其中与最优化理论最相关的概念我认为是证据下界（Evidence Lower Bound，ELBO），这是一种用于变分推断的量，用于近似模型的对数边际似然。在变分自编码器（Variational Autoencoder，VAE）中，ELBO 被用作优化模型参数的目标函数。ELBO 由两个部分组成：重构损失和先验分布与近似后验分布之间的 KL 散度。通过最大化 ELBO 来学习生成模型，从而可以从所学习的分布中生成新的数据样本。

$$\log p(x) \geq \mathbb{E}_{q(z|x)}[\log p(x|z)] - D_{KL}(q(z|x)||p(z))$$

其中，x 是输入数据，z 是潜在变量，$p(x)$ 是真实数据分布，$q(z|x)$ 是给定 x 时的近似后验分布，$p(z)$ 是先验分布。左侧为对数边际似然下界（ELBO），右侧第一项为重构损失（reconstruction loss），衡量了模型重构输入数据的能力；右侧第二项为 KL 散度（Kullback-Leibler divergence），衡量了近似后验分布与先验分布之间的差异。

扩散模型的关键在于训练噪声预测模型。为了提高模型性能，我们可以利用最优化算法对扩散模型进行改进。以下是一些可能的改进方向：

1. 优化目标函数：在训练扩散模型时，可以尝试使用不同的目标函数，如最小化重建误差、最大化似然等。通过优化目标函数，我们可以使模型更好地捕捉数据的统计规律，从而提高生成样本的质量。

2. 优化网络结构：扩散模型的去噪过程通常使用 U-net 实现。我们可以尝试使用不同的网络结构，如 transformer 等，以提高模型的性能。此外，可以考虑引入注意力机制、跳跃连接等技术，以进一步改善模型的性能。

3. 优化训练策略：在训练扩散模型时，可以尝试使用不同的优化算法，如梯度下降法、随机梯度下降法、Adam 等。此外，可以调整学习率、批次大小等超参数，以提高模型的收敛速度和性能。