零点

二分法

$$x_{n+1} - x^* = \frac{x_0 - x^*}{2^{n+1}}$$

牛顿法

$$x_{n+1} = \phi(x_n) = x_n - \frac{f(x_n)}{f^{(1)}(x_n)}$$

收敛速度判断

由于

$$
\begin{aligned}
\phi(x) &= x - \frac{f(x)}{f^{(1)}(x)} \\
\phi^{(1)}(x) &= 1 - \frac{f^{(1)}(x)^2 - f(x)f^{(2)}(x)}{f^{(1)}(x)^2} \\
&= \frac{f(x)f^{(2)}(x)}{f^{(1)}(x)^2}
\end{aligned}
$$

且在零点 x^* 处

$$f(x^*) = 0$$

得

$$\phi^{(1)}(x^*) = 0$$

根据泰勒展开，将$\phi(x_n)$在零点x^*展开

$$\phi(x_n) = \phi(x^*) + \phi^{(1)}(x^*)(x_n - x^*) + \frac{1}{2!}\phi^{(2)}(x^*)(x_n - x^*)^2 + \ldots$$

且

$$
\begin{aligned}
x_{n+1} &= \phi(x_n) \\
x^* &= \phi(x^*) \\
\phi^{(1)}(x^*) &= 0
\end{aligned}
$$

得

$$
\begin{aligned}
x_{n+1} - x^* &= \phi(x_n) - \phi(x^*) \\
&\approx \phi^{(2)}(x^*)(x_n - x^*)^2 \\
&\approx \phi^{(2)}(x^*)(\phi^{(2)}(x^*)(x_{n-1} - x^*)^2 - x^*)^2 \\
&= \ldots\ldots \\
&\approx (x_0 - x^*)^{2^{n+1}}
\end{aligned}
$$

即

$$x_{n+1} - x^* \approx (x_0 - x^*)^{2^{n+1}}$$

所以，牛顿法满足2阶收敛，收敛速度较快

重根问题

$f(x)$有m个重根，则$f(x)$可写作

$$f(x) = (x - x^*)^m$$

且

$$f^{(1)}(x^*) = f^{(2)}(x^*) = \ldots = f^{(m-1)}(x^*) = 0$$

此时，对于$\phi(x)$，有

$$
\begin{aligned}
\phi(x) &= x - \frac{f(x)}{f^{(1)}(x)} \\
&= x - \frac{x - x^*}{m} \\
\phi^{(1)}(x) &= 1 - \frac{1}{m}
\end{aligned}
$$

由于$\phi^{(1)}(x)$不为0，牛顿法退化为线性收敛(二分法)

改进

可改写递推式：

$$
\begin{aligned}
\phi(x) = x - m\frac{f(x)}{f^{(1)}(x)} = x^* \\
\phi^{(1)}(x) = 0
\end{aligned}
$$

方程组求解

高斯消去法

上三角矩阵U与下三角矩阵L

计算

$$
\begin{aligned}
Ux = b \\
x_k = b_k - \sum_{j=k+1}^{n} U_{k,j}x_j
\end{aligned}
$$

```
for (int k = n; 1 <= k; k--)
    for (int j = k+1; j <= n; j++)
        x[k] = b[k] - U[k][j] * x[j]
```

计算

$$
\begin{aligned}
Lx = b \\
x_k = b_k - \sum_{j=1}^{k-1} L_{k,j}x_j
\end{aligned}
$$

```
for (int k = 1; k <= n; k++)
    for (int j = 1; j <= k-1; j++)
        x[k] = b[k] - L[k][j] * x[j]
```

原理

对于

$$Ax = b$$

使用高斯变换，乘以下三角矩阵L

$$LAx = Lb$$
$$Ux = Lb$$
$$x = U^{-1}Lb$$

因此关键在于，将系数矩阵A转化为上三角矩阵U

$$LA = U$$

即，将矩阵A分解为下三角矩阵L和上三角矩阵之积U

$$A = LU$$

称为矩阵LU分解

$$a_{k,p} = a_{k,p} - \frac{a_{k,j}}{a_{j,j}} * a_{j,p}$$

```
gauss_lu(double* a, int n)--> void {
    for(j = 1; j <= n; ++j){
        for(k = j+1; k <= n; ++k){
            temp = a[k][j] = a[k][j]/a[j][j];
            for(p =j+1; p <= n; ++p)
                a[k][p] = a[k][p] - a[j][p] * temp;
        }
    }
}
```

列选主元高斯消去法

交换行：选取一列中最大值，然后交换。避免很大的数字对结果产生影响

```
guass_lu(double* a, int* p, int n)--> void
{
    p[1,n] = {1, 2, · · · , n};
    for(j = 1; j <= n; ++j) {
        // 找到a[j][j]到a[n][j]的最大值a[t][j]
        swap(a[j], a[t]);
        swap(p[j], p[t]);
        for(k = j+1; k <= n; ++k) {
            temp = a[k][j] = a[k][j]/a[j][j];
            for(s = j+1; s <= n; ++s)
                a[k][s] = a[k][s] - a[j][s] * temp;
        }
}
```

完全选主元高斯消去法

交换行和列

误差分析

具体内容见讲义

前向误差

由高斯法求解得到x的近似值

$$x^*$$

判断相对误差

$$\frac{\delta(x)}{x} = \frac{x^* - x}{x}$$

的上界

向后舍入误差分析

存在ΔA满足

$$(A + \Delta A)x^* = b$$

上三角矩阵方程组数组解的舍入误差分析

$$|\Delta b| = |Ux^* - b|$$

列选主元高斯消去法*LU*分解的舍入误差分析

$$|\Delta A| = |A - LU|$$

完全选主元高斯消去法*LU*分解的舍入误差分析

$$|\Delta A| = |A - LU|$$

列选主元高斯消去法解方程组舍入误差分析

$$|\Delta b| = |Ax^* - b|$$

范数

具体内容见讲义

将向量和矩阵映射到实数范围，比较大小

向量范数

壹范数、贰范数、*p*范数、无穷范数

矩阵范数

行范数(无穷范数)、列范数(壹范数)、贰范数(欧几里得范数、谱范数)、*p*范数、*F*范数

迭代法

计算

$$Ax = b$$

使用迭代法

收敛判断

对于

$$x = Gx + b$$
$$\begin{cases} x^0 = \text{随机选取} \\ x^{k+1} = Gx^k + b \end{cases}$$

上面迭代格式的充要条件是$\rho(G) < 1$

Jacobi迭代

$$A = L + U + D$$
$$x = -D^{-1}(L+U)x + D^{-1}b$$

计算格式

$$x_0 = \text{初始向量}$$
$$x_{k+1} = D^{-1}(b - (L+U)x_k), k = 0, 1, 2...$$

如果A主对角线元行占优(或列占优)，则上面迭代格式收敛

Gauss-Seidel迭代

$$A = L + U + D$$
$$(L+D)x = -Ux + b$$

计算格式

$$x_0 = \text{初始向量}$$
$$x_{k+1} = D^{-1}(b - Ux_k - Lx_{k+1}), k = 0, 1, 2...$$

如果A

1. 主对角线元占优；或
2. 对称正定

则上面迭代格式收敛

代码

```
见讲义
```

逐次松弛迭代

使用*Gauss-Seidel*迭代的求解方向，并使用过松弛以加快收敛速度

令*w*是一个实数，将x_{k+1}定义为*w*乘上*Gauss-Seidel*公式和$1-w$乘上当前估计x_k的平均。

*w*称为松弛参数，$w>1$时称为过松弛

$$Dx_{k+1}=(1-w)Dx_k+w(b-Lx_{k+1}-Ux_k)$$

计算格式

$$\begin{gathered}x_0=\text{初始向量}\\ x_{k+1}=(wL+D)^{-1}[(1-w)Dx_k-wUx_k]+w(D+wL)^{-1},b,k=0,1,2...\end{gathered}$$

如果*A*

1. 主对角线占优并且$w\epsilon(0,1]$；或者
2. 对称正定并且$w\epsilon(0,2)$

则上面迭代格式收敛

代码

```
见讲义
```

用于对称正定矩阵的方法

对称正定矩阵

对称矩阵只有一半数量的独立元素

$$A^T=A$$

是否能以一半的计算代价，并且仅仅使用一半的内存来求解

楚列斯基分解

如果*A*是对称正定矩阵，则存在上三角矩阵*R*满足

$$A=R^TR$$

共轭梯度方法

共轭

定义A内积

$$(v,w)_A = v^T A w$$

当$(v,w)_A = 0$时，向量v和w为A共轭

计算格式

$$\begin{gathered} x_0 = \text{初始估计} \\ p_0 = r_0 = b - Ax_0 \\ for\ k = 0,1,2,\dots \\ \alpha_k = \frac{r_k^T r_k}{p_k^T A p_k} \\ x_{k+1} = x_k + \alpha_k p_k \\ r_{k+1} = r_k - \alpha_k A p_k \\ \beta_k = \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k} \\ p_{k+1} = r_{k+1} + \beta_k p_k \end{gathered}$$

向量x_k是第k步时的近似解. 向量r_k表示近似解x_k的余项

变量p_k表示用于更新x_k得到改进的x_{k+1}时所使用的新的搜索方向

对于α_k和β_k的选择

1. 选择α_k使得新的余项r_{k+1}和方向p_k正交，保证下一余项向量和前面所有的余项向量都正交，即$(r^{k+1}, r^k) = 0$

   $r^0, r^1, \dots, r^{n-1}$是正交的
2. 选择β_k，保证p_{k+1}和$p_k$$A$共轭，即$(p_{k+1}, p_k)_A = 0$

   $p^0, p^1, \dots, p^{n-1}$是A共轭的

代码

```
见讲义
```

最小二乘

最小二乘解

法线方程

考虑方程组

$$Ax = b$$

当解不存在时，可以试着找到近似解，即最小二乘近似

方程组可写为

$$x_1 v_1 + x_2 v_2 + \ldots + x_n v_n = b$$

可以把b看作是A的列向量v_i的线性组合，对应的组合系数是$x_1, \ldots, x_n$

即向量$v_1, v_2, \ldots, v_n$生成的空间中找到组合系数$x_1, \ldots, x_n$，使b在这个空间上

当解不存在时，在向量空间A={$v_1, v_2, \ldots, v_n$}中存在与b最接近的点：
即存在$\overline{x}$，满足

$$b - A\overline{x} \perp \text{平面}\{Ax|x\epsilon R^n\}$$

把垂直性表示为矩阵的乘法，对于R^n上所有的x，

$$(Ax)^T(b - A\overline{x}) = 0$$

可写为

$$x^T A^T (b - A\overline{x}) = 0$$

这意味着n维向量$A^T(b - A\overline{x})$和R^n中其他的n维向量垂直，这表明

$$A^T(b - A\overline{x}) = 0$$

即

$$A^T A\overline{x} = A^T b$$

称为法线方程，它的解$\overline{x}$是方程组$Ax = b$的最小二乘解

余项

最小二乘解$\overline{x}$的余项

$$r = b - A\overline{x}$$

大小度量：

- 向量的欧氏长度(2范数)

$$\| r \|_2 = \sqrt{r_1^2 + \cdots + r_m^2}$$

- 平方误差

$$SE = r_1^2 + \cdots + r_m^2$$

- 平均平方根误差

$$RMSE = \sqrt{\frac{SE}{m}} = \sqrt{\frac{r_1^2 + \cdots + r_m^2}{m}}$$

QR分解

Gram-Schmidt正交

对一组向量正交化：给定一组输入的m维向量，找出正交坐标系统，获得由这些向量张成的空间

对于A向量和B向量，找到正交单位向量：

$$q_1 = A$$
$$q_1 = \frac{q_1}{\|q_1\|_1}$$

接着找到B向量在A向量上的投影，然后用B向量减去该投影即可得到与A向量垂直的向量：

$$q_2 = B - q_1(q_1^T B)$$

然后转化为单位向量

$$q_2 = \frac{q_2}{\|q_2\|_2}$$

推广到多维向量，令$A_1, \ldots, A_n$是R^m中的线性无关向量：

$$y_j = A_j - q_1(q_1^T A_j) - q_2(q_2^T A_j) - \cdots - q_{j-1}(q_{j-1}^T A_j)$$
$$q_j = \frac{y_j}{\|y_j\|_2}$$

对于上面的结果，引入新的符号：

$$r_{jj} = \| y_j \|_2$$
$$r_{ij} = q_i^T A_j$$

则

$$A_1 = r_{11} q_1$$
$$A_2 = r_{12} q_1 + r_{22} q_2$$
$$\cdots$$
$$A_j = r_{1j} q_1 + \cdots + r_{j-1,j} q_{j-1} + r_{jj} q_j$$

即

$$A_{m \times n} = Q_{m \times n} R_{n \times n}$$

称为消减QR分解

$$A_{m \times n} = Q_{m \times m} R_{m \times n}$$

称为完全QR分解

实现最小二乘：

误差向量e满足

$$e = \| Ax - b \|_2 = \| QRx - b \|_2 = \| Rx - Q^T b \|_2$$

解$\overline{x}$即为最小二乘解

改进：

$$y = A_j - q_1(q_1^T y) - q_2(q_2^T y) - \cdots - q_{j-1}(q_{j-1}^T y)$$

Household变换

*Household*反射子是正交矩阵，通过$m-1$维平面反射m维向量。

即给定一个向量x，重新找出一个相同长度的向量w，计算*Household*反射得出矩阵H满足

$$Hx = w$$

定义向量$v = w - x$，考虑投影矩阵

$$P = \frac{vv^T}{v^T v}$$

对于任何向量u，Pu是u在v上的投影

令$H = I - 2P$，则

$$Hx = x - 2Px = w - v - \frac{2vv^T}{v^T v}x = w - \frac{vv^T(w+x)}{v^T v} = w$$

矩阵H被称为**Household反射子**

实现QR分解：

$$H_3 H_2 H_1 A = R$$
$$A = H_1 H_2 H_3 R = QR$$