May 14, 2014

Symmetric positive definite systems

Theorem: If A is SPD, then there exists unique lower Δ G such that

$$A = G G^T$$

Pf: $A = L D L^T$ and $D = \text{diag}(d_1, \ldots, d_n)$, $d_i > 0$.

Define $D^{1/2} = \text{diag}(\text{sqrt}\{d_1\}, \ldots, \text{sqrt}\{d_n\})$

Let $G = L D^{1/2}$. Then G is lower Δ.

$$G G^T = L D^{1/2} (L D^{1/2})^T = L D^{1/2} D^{1/2} L^T = L D L^T = A$$

- $A = G G^T$ is called the Cholesky factorization of A and the lower Δ G is called the Cholesky factor.

Cholesky factorization

$$A = \begin{bmatrix} \alpha & v^T \\ v & B \end{bmatrix} = \begin{bmatrix} \sqrt{\alpha} & 0 \\ \frac{v}{\sqrt{\alpha}} & I \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & B - \frac{vv^T}{\alpha} \end{bmatrix} \begin{bmatrix} \sqrt{\alpha} & \frac{v^T}{\sqrt{\alpha}} \\ 0 & I \end{bmatrix}$$

Let $X = \begin{bmatrix} 1 & -\frac{v^T}{\alpha} \\ 0 & I \end{bmatrix}$. Then X has full rank.

Also $B - (vv^T)/\alpha = X^T A X \Rightarrow$ SPD

Hence $B - (vv^T)/\alpha = G_1 G_1^T$. Now define

$$G = \begin{bmatrix} \sqrt{\alpha} & 0 \\ \frac{v}{\sqrt{\alpha}} & G_1 \end{bmatrix}$$

Then $A = G G^T$.

Algorithm

for $k = 1, 2, \dots, n$
    $a_{kk} = \text{sqrt}\{a_{kk}\}$
    for $j = k+1, \dots, n$
        $a_{jk} = a_{jk} / a_{kk}$
    end
    for $j = k+1, \dots, n$
        for $i = j, \dots, n$
            $a_{ij} = a_{ij} - a_{ik} a_{jk}$
        end
    end
end

- flops(Cholesky) ~ $n^3/3$

Banded systems

Def: A has upper bandwidth q if $a_{ij} = 0$, $j > i+q$, and lower bandwidth p if $a_{ij} = 0$, $i > j+p$:

$$
\begin{array}{cc}
 & \overbrace{}^{q+1} \\
A = \begin{array}{r} p+1\left\{\phantom{\begin{matrix}x\\x\end{matrix}}\right. \\ \\ \end{array} &
\begin{bmatrix} & & \ddots & 0 \\ & & & \ddots \\ \ddots & & & \\ 0 & \ddots & & \end{bmatrix}
\begin{array}{l} \\ \\ \left.\phantom{\begin{matrix}x\\x\end{matrix}}\right\} q+1 \end{array} \\
 & \underbrace{}_{p+1}
\end{array}
$$

If A is banded, so are L U, G G^T, and L D M^T.

Theorem: Let A = L U. If A has upper bandwidth q and lower bandwidth p, then U has upper bandwidth q and L has lower bandwidth p.

$$
p\left\{\phantom{\begin{matrix}x\\x\end{matrix}}\right.
\overset{\overbrace{}^{q}}{\begin{bmatrix} & & \ddots & \\ & & & \ddots \\ \ddots & & & \\ & \ddots & & \end{bmatrix}}
=
p\left\{\phantom{\begin{matrix}x\\x\end{matrix}}\right.
\begin{bmatrix} \ddots & & & \\ & \ddots & & \\ \ddots & & \ddots & \\ & \ddots & & \ddots \end{bmatrix}
\begin{bmatrix} \ddots & & \ddots & \\ & \ddots & & \ddots \\ & & \ddots & \\ & & & \ddots \end{bmatrix}
\left.\phantom{\begin{matrix}x\\x\end{matrix}}\right\}q
$$

$$
\qquad A \qquad\qquad\qquad L \qquad\qquad\qquad U
$$

Algorithm (Band GE)

```
for  k = 1, 2, … , n-1
    for  i = k+1, … , min(k+p, n)
        a_ik = a_ik / a_kk
    end
    for  i = k+1, … , min(k+p, n)
        for  j = k+1, … , min(k+q, n)
            a_ij = a_ij – a_ik a_kj
        end
    end
end
```

If $n >> p$ and $n >> q$, then flops(band GE) ~ $2\,n\,p\,q$

Exercise: band forward / back solves

Tridiagonal systems (skipped)

- Here, we also assume A is symmetric.

$$L = \begin{bmatrix} 1 & & & 0 \\ e_1 & \ddots & & \\ & \ddots & \ddots & \\ 0 & & e_{n-1} & 1 \end{bmatrix} \qquad D = \begin{bmatrix} d_1 & & & \\ & d_2 & & \\ & & \ddots & \\ & & & d_n \end{bmatrix}$$

$A = L\,D\,L^T$ =>

$$
\begin{aligned}
a_{kk} &= (LDL^T)_{kk} \\
&= \sum_i \sum_j l_{ki}\ d_{ij}\ l^T_{jk} \\
&= \sum_i l_{ki}\ d_{ii}\ l^T_{ik} = \sum_i l^2_{ik}\ d_{ii} \\
&= l^2_{k,k-1}\ d_{k-1,k-1} + l^2_{kk}\ d_{kk} \qquad (i = k-1, k) \\
&= e^2_{k-1}\ d_{k-1} + d_k \\
a_{k,k-1} &= (LDL^T)_{k,k-1} \\
&= \sum_i \sum_j l_{ki}\ d_{ij}\ l^T_{j,k-1} \\
&= \sum_i l_{ki}\ d_{ii}\ l^T_{i,k-1} = \sum_i l_{ki}\ d_{ii}\ l_{k-1,i} \\
&= l_{k,k-1}\ d_{k-1,k-1}\ l_{k-1,k-1} \qquad (i = k-1, k) \\
&= e_{k-1}\ d_{k-1}
\end{aligned}
$$

Algorithm

$d_1 = a_{11}$
for $k = 2, \ldots, n$
  $e_{k-1} = a_{k,k-1} / d_{k-1}$
  $d_k = a_{kk} - e_{k-1}\, a_{k,k-1}$
end

- flops(tridiag) = O(n)

Sparse Matrices

(1) Usually a constant number of nonzeros per row

i.e. $O(n)$ number of nonzero entries

- store only the nonzero entries

(2) In GE/LU, the main computation:

$$a_{ij} = a_{ij} - a_{ik}\, a_{kj} / a_{kk}$$

$$= 0 - 0 \times 0 \qquad\qquad \text{(most entries are 0)}$$

- never operate on zero's

(3) A is sparse, but L, U can be dense

e.g. "Arrow" matrix

$$A = \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & & & \\ \times & & \times & & \\ \times & & & \times & \\ \times & & & & \times \end{bmatrix} = \begin{bmatrix} 1 & & & & \\ \times & 1 & & 0 & \\ \times & \times & \ddots & & \\ \times & \times & \times & \ddots & \\ \times & \times & \times & \times & 1 \end{bmatrix} \begin{bmatrix} \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & & \times & \times & \times \\ & 0 & & \times & \times \\ & & & & \times \end{bmatrix}$$

The storage for L & U = $O(n^2)$, computation of LU = $O(n^3)$

- use a different ordering of unknowns:

$$\tilde{x}_1 = x_2,\ \ \tilde{x}_2 = x_3,\ \ \ldots,\ \ \tilde{x}_n = x_1$$

Then

$$A = \begin{bmatrix} \times & & & & \times \\ & \times & & & \times \\ & & \times & & \times \\ & & & \times & \times \\ \times & \times & \times & \times & \times \end{bmatrix} = \begin{bmatrix} \times & & & & \\ & \times & & 0 & \\ & & \times & & \\ & 0 & & \times & \\ \times & \times & \times & \times & \times \end{bmatrix} \begin{bmatrix} \times & & & & \times \\ & \times & & 0 & \times \\ & & \times & & \times \\ & 0 & & \times & \times \\ & & & & \times \end{bmatrix}$$

Application problem: heat conduction

Heat conduction can be modelled by a partial differential equation (PDE):

$$-\left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} + \frac{\partial^2 T}{\partial z^2}\right) = f(x,y,z)$$

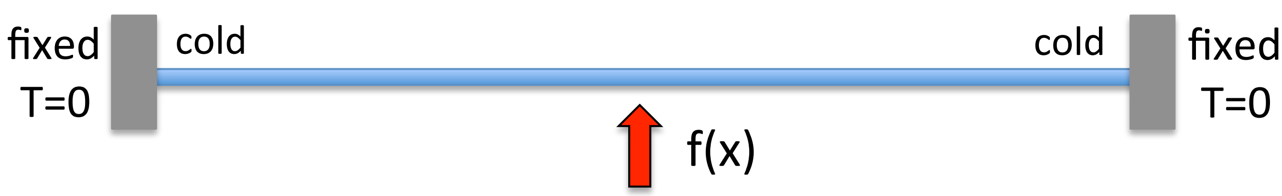
where x, y, z = coordinates over a known range

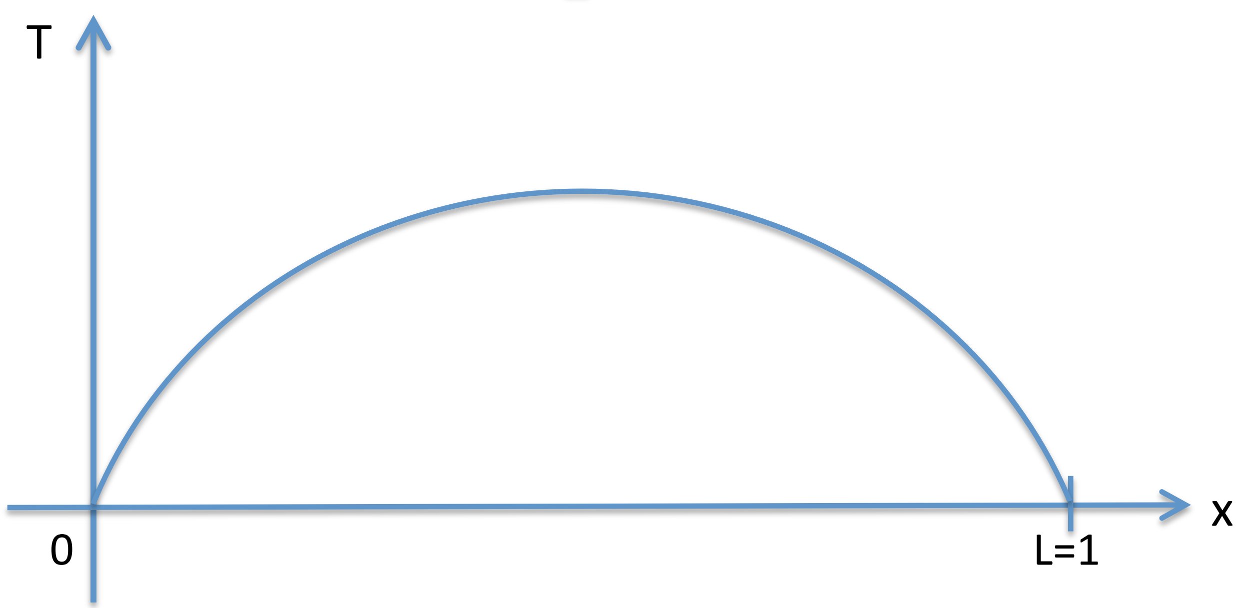
T(x,y,z) = temperature at (x,y,z)

f(x,y,z) = source function

One dimension

$$-\frac{\partial^2 T}{\partial x^2} = f(x)$$





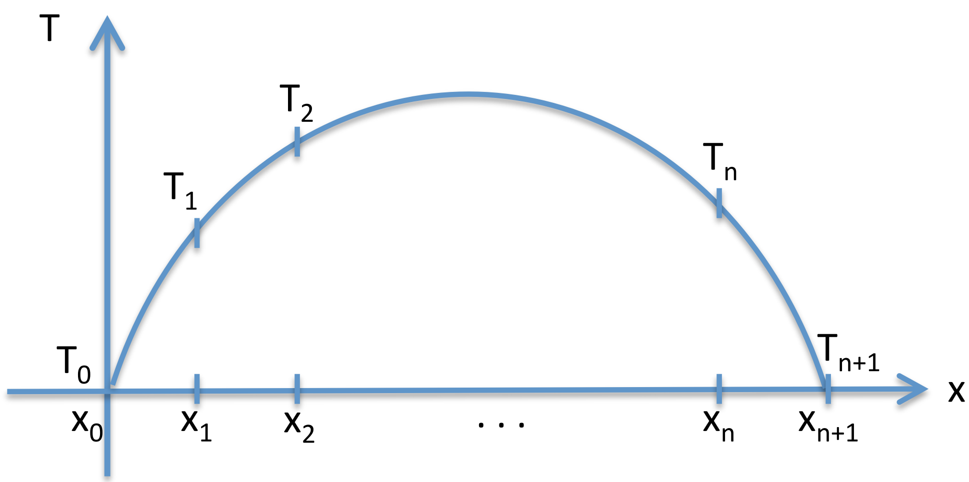
How to compute T?

Divide the interval [0, 1] into subintervals:

$$0 = x_0 < x_1 < x_2 \dots < x_{n+1} = 1$$

$\{ x_i \}$ are called grid points.

Approximate the temperature T at x_i : $T_i \approx T(x_i)$



Notes

(1) We assume temp = 0 at both ends

i.e. $T_0 = T_{n+1} = 0$

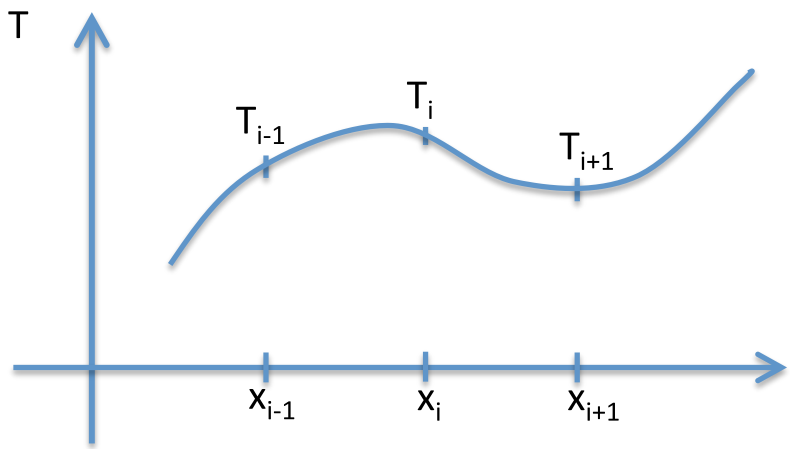
Thus the unknowns are: $T_1, T_2, \dots, T_n$

(2) Uniform spacing:

$$h = x_i - x_{i-1} = 1/(n+1)$$

= grid size / mesh size

Finite difference approximation



$$\frac{\partial T}{\partial x}\left(x_i^-\right) \approx \frac{T_i - T_{i-1}}{h} \qquad \text{(backward differencing)}$$

$$\frac{\partial T}{\partial x}\left(x_i^+\right) \approx \frac{T_{i+1} - T_i}{h} \qquad \text{(forward differencing)}$$

$$\frac{\partial^2 T}{\partial x^2}\left(x_i\right) \approx \frac{\frac{\partial T}{\partial x}\left(x_i^+\right) - \frac{\partial T}{\partial x}\left(x_i^-\right)}{h}$$

$$= \frac{\frac{T_{i+1} - T_i}{h} - \frac{T_i - T_{i-1}}{h}}{h}$$

$$= \frac{T_{i-1} - 2T_i + T_{i+1}}{h} \qquad \text{(central differencing)}$$

For each x_i, $i = 1, 2, \ldots, n$, we have one equation:

$$-\frac{T_{i-1} - 2T_i + T_{i+1}}{h} = f_i \qquad (f_i = f(x_i))$$

$$i.e. \quad -\frac{1}{h^2} T_{i-1} + \frac{2}{h^2} T_i - \frac{1}{h^2} T_{i+1} = f_i \qquad i = 1, 2, \ldots, n$$

=> a system of linear equations!