Calculus, finite differences Interpolation, Splines, NURBS

CMSC 828 D

Least Squares, SVD, Pseudoinverse

- $\mathbf{Ax}=\mathbf{b}$ $\mathbf{A}$ is $m\times n$, $\mathbf{x}$ is $n\times 1$ and $\mathbf{b}$ is $m\times 1$.
- $\mathbf{A}=\mathbf{USV}^t$ where $\mathbf{U}$ is m×m, $\mathbf{S}$ is m×n and $\mathbf{V}$ is $n\times n$
- $\mathbf{USV}^t\mathbf{x}=\mathbf{b}$. So $\mathbf{SV}^t\mathbf{x}=\mathbf{U}^t\mathbf{b}$
- If $\mathbf{A}$ has rank r, then r singular values are significant

  $\mathbf{V}^t\mathbf{x}=\mathrm{diag}(\sigma_1^{-1},\ldots,\sigma_r^{-1},0,\ldots,0)\mathbf{U}^t\mathbf{b}$

  $\mathbf{x}=\mathbf{V}\mathrm{diag}(\sigma_1^{-1},\ldots,\sigma_r^{-1},0,\ldots,0)\mathbf{U}^t\mathbf{b}$

$$\mathbf{x}_r=\sum_{i=1}^{r}\frac{\mathbf{u}_i^t\mathbf{b}}{\sigma_i}\mathbf{v}_i \qquad \sigma_r>\varepsilon,\ \ \sigma_{r+1}\le\varepsilon$$

•Pseudoinverse $\mathbf{A}^+=\mathbf{V}\,\mathrm{diag}(\sigma_1^{-1},\ldots,\sigma_r^{-1},0,\ldots,0)\,\mathbf{U}^t$

- $\mathbf{A}^+$ is a $n\times m$ matrix.
- If rank $(\mathbf{A})=n$ then $\mathbf{A}^+=(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{A}$
- If $\mathbf{A}$ is square $\mathbf{A}^+=\mathbf{A}^{-1}$

Well Posed problems

- Hadamard postulated that for a problem to be "well posed"
  1. Solution must exist
  2. It must be unique
  3. Small changes to the input data should cause small changes to the solution
- Many problems in science and computer vision result in "ill-posed" problems.
  - Numerically it is common to have condition 3 violated.
- Recall from the SVD $\mathbf{x}=\sum_{i=1}^{n}\frac{\mathbf{u}_i^t\mathbf{b}}{\sigma_i}\mathbf{v}_i \qquad \sigma_r>\varepsilon,\ \ \sigma_{r+1}\le\varepsilon$

•If σs are close to zero small changes in the "data" vector $\mathbf{b}$ cause big changes in $\mathbf{x}$.

•Converting ill-posed problem to well-posed one is called *regularization*.

Regularization

- Pseudoinverse provides one means of regularization
- Another is to solve $(\mathbf{A}+\varepsilon\mathbf{I})\mathbf{x}=\mathbf{b}$ $\quad \mathbf{x}=\sum_{i=1}^{n}\frac{\sigma_i}{\varepsilon+\sigma_i^2}(\mathbf{u}_i^t\mathbf{b})\mathbf{v}_i$

•Solution of the regular problem requires minimizing of $\|\mathbf{Ax}-\mathbf{b}\|^2$

•This corresponds to minimizing

$$\|\mathbf{Ax}-\mathbf{b}\|^2+\varepsilon\|\mathbf{x}\|^2$$

- Philosophy – pay a "penalty" of O(ε) to ensure solution does not blow up.
- In practice we may know that the data has an uncertainty of a certain magnitude … so it makes sense to optimize with this constraint.

•Ill-posed problems are also called "ill-conditioned"

Outline

- Gradients/derivatives
  - needed in detecting features in images
    - Derivatives are large where changes occur
  - essential for optimization
- Interpolation
  - Calculating values of a function at a given point based on known values at other points
  - Determine error of approximation
  - Polynomials, splines
- Multiple dimensions

Derivative

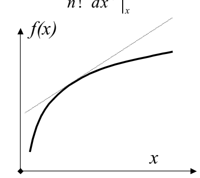
- In 1-D $\quad \frac{df}{dx}=\lim_{h\to 0}\frac{f(x+h)-f(x)}{h}$
- Taylor series: for a continuous function

$$f(x+h)=f(x)+h\left.\frac{df}{dx}\right|_x+\frac{h^2}{2}\left.\frac{d^2f}{dx^2}\right|_x+\cdots+\frac{h^n}{n!}\left.\frac{d^nf}{dx^n}\right|_x+\cdots$$

$$f(x-h)=f(x)-h\left.\frac{df}{dx}\right|_x+\frac{h^2}{2}\left.\frac{d^2f}{dx^2}\right|_x+\cdots+(-1)^n\frac{h^n}{n!}\left.\frac{d^nf}{dx^n}\right|_x+\cdots$$

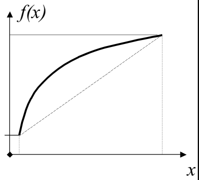
•Geometric interpretation

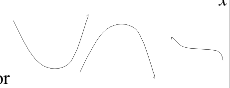
- Approximate smooth curve by values of tangent, curvature, etc.



Remarks

- Mean value theorem:
  - $f(b)-f(a)=(b-a)df/dx|_c$ $a<c<b$
  - There is at least one point between a and b on the curve where the slope matches that of the straight line joining the two points





•$df/dx=0$

–represents a minimum, maximum or saddle point of the curve $y=f(x)$

–$d^2f/dx^2 > 0$ minimum, $d^2f/dx^2 < 0$ maximum

–$d^2f/dx^2 = 0$ saddle point

Finite differences

- Approximate derivatives at points by using values of a function known at certain neighboring points
- Truncate Taylor series and obtain an expression for the derivatives
- Forward differences: use value at the point and forward x —— x —— x —— x
- Backward differences

$$\left.\frac{df}{dx}\right|_x = h^{-1}\left(f(x+h)-f(x)\right)-\frac{h}{2}\left.\frac{d^2f}{dx^2}\right|_x + O\left(h^2\right)$$

$$\left.\frac{df}{dx}\right|_x = h^{-1}\left(f(x)-f(x-h)\right)+\frac{h}{2}\left.\frac{d^2f}{dx^2}\right|_x + O\left(h^2\right)$$

Finite Differences

- Central differences
  - Higher order approximation

$$2\left.\frac{df}{dx}\right|_x = \frac{f(x+h)-f(x)}{h}-\frac{h}{2}\left.\frac{d^2f}{dx^2}\right|_x + \frac{f(x)-f(x-h)}{h}+\frac{h}{2}\left.\frac{d^2f}{dx^2}\right|_x + O\left(h^2\right)$$

$$\left.\frac{df}{dx}\right|_x = \frac{f(x+h)-f(x-h)}{2h}+O\left(h^2\right)$$

–However we need data on both sides

–Not possible for data on the edge of an image

–Not possible in time dependent problems (we have data at current time and previous one)

Approximation

- Order of the approximation $O(h)$, $O(h^2)$
- Sidedness, one sided, central etc.
- Points around point where derivative is calculated that are involved are called the “stencil” of the approximation.
- Second derivative

$$0 = \frac{f(x+h)-f(x)}{h}-\frac{h}{2}\left.\frac{d^2f}{dx^2}\right|_x - \frac{f(x)-f(x-h)}{h}+\frac{h}{2}\left.\frac{d^2f}{dx^2}\right|_x + O\left(h^2\right)$$

$$\frac{d^2f}{dx^2} = \frac{f(x+h)-2f(x)+f(x-h)}{h^2}+O(h)$$

•One sided difference of $O(h^2)$

$$\frac{df}{dx} = \frac{-3f(x)+4f(x+h)-f(x+2h)}{2h}+O(h^2)$$

Polynomial interpolation

- Instead of playing with Taylor series we can obtain fits using polynomial expansions.
  - 3 points fit a quadratic ax^2+bx+c
    - Can calculate the 1st and 2nd derivatives
  - 4 points fit a cubic, etc.
- Given x_1, x_2, x_3, x_4 and values f_1, f_2, f_3, f_4

$$\begin{bmatrix}1 & x_1 & x_1^2 & x_1^3\\ 1 & x_2 & x_2^2 & x_2^3\\ 1 & x_3 & x_3^2 & x_3^3\\ 1 & x_4 & x_4^2 & x_4^3\end{bmatrix}\begin{bmatrix}a_0\\ a_1\\ a_2\\ a_3\end{bmatrix}=\begin{bmatrix}f_1\\ f_2\\ f_3\\ f_4\end{bmatrix} \qquad \begin{bmatrix}a_0\\ a_1\\ a_2\\ a_3\end{bmatrix}=\begin{bmatrix}1 & x_1 & x_1^2 & x_1^3\\ 1 & x_2 & x_2^2 & x_2^3\\ 1 & x_3 & x_3^2 & x_3^3\\ 1 & x_4 & x_4^2 & x_4^3\end{bmatrix}^{-1}\begin{bmatrix}f_1\\ f_2\\ f_3\\ f_4\end{bmatrix}$$

•Vandermonde system – fast algorithms for solution.

•If more data than degree .. Can get a least squares solution.

•Matlab functions `polyfit, polyval`

Remarks

- Can use the fitted polynomial to calculate derivatives
- If equation is solved analytically this provides expressions for the derivatives.
- Equation can become quite ill conditioned
  - especially if equations are not normalized. ax^2+bx+c can also be written as $a^*(x-x_0)^2+b^*(x-x_0)+c^*$
  - *Find the polynomial through* x_0-h, x_0, x_0+h

$$\begin{bmatrix}1 & -h & h^2\\ 1 & 0 & 0\\ 1 & h & h^2\end{bmatrix}\begin{bmatrix}a_0\\ a_1\\ a_2\end{bmatrix}=\begin{bmatrix}f_{-1}\\ f_0\\ f_1\end{bmatrix} \qquad \begin{bmatrix}1 & -h & h^2\\ 1 & 0 & 0\\ 1 & h & h^2\end{bmatrix}^{-1}=\begin{bmatrix}0 & 1 & 0\\ -\frac{1}{2h} & 0 & \frac{1}{2h}\\ \frac{1}{2h^2} & -\frac{1}{h^2} & \frac{1}{2h^2}\end{bmatrix}$$

–$a_0=f_0$, $a_1=(f_1-f_{-1})/2h$ $a_2=(f_{-1}-2f_0+f_1)/2h^2$

–Gives the expected values of the derivatives.

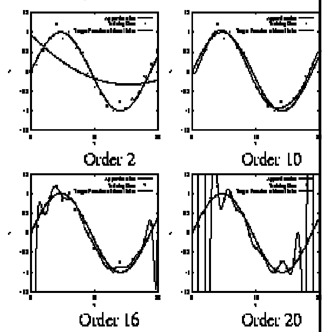
Polynomial interpolation

- Results from Algebra
  - Polynomial of degree n through $n+1$ points is unique
  - Polynomials of degree less than x^n is an n dimensional space.
  - $1, x, x^2, \dots, x^{n-1}$ form a basis.
    - Any other polynomial can be represented as a combination of these basis elements.
  - Other sets of independent polynomials can also form bases.
- To fit a polynomial through $x_0, \dots, x_n$ with values $f_0, \dots, f_n$
  - Use Lagrangian basis l_k. $l_k = \prod_{\substack{i=0,\\ i\neq k}}^{n} \frac{x - x_i}{x_k - x_i}, \quad k = 0, \dots, n$

–$p(x)=a_0 l_0 + a_1 l_1 + \dots + a_n l_n$.
–Then $a_i = f_i$
–Many polynomial bases: Chebyshev, Legendre, Laguerre …
–Bernstein, Bookstein …

Increasing *n*

- As n increases we can increase the polynomial degree.
- However the function in between is very poorly interpolated.
- Becomes ill-posed.
- For large n *interpolant blows up.*



•Idea:
–Taylor series provides good local approximations
–Use local approximations
•Splines

Spline interpolation

- Piecewise polynomial approximation
  - E.g. interpolation in a table
  - Given x_k, x_{k+1}, f_k *and* f_{k+1} evaluate f at a point x such that $x_k < x < x_{k+1}$

$$f(x) = \begin{cases} f_{k+1}\dfrac{x - x_k}{x_{k+1} - x_k} + f_k \dfrac{x - x_{k+1}}{x_k - x_{k+1}}, & x_k \le x \le x_{k+1} \\ 0 & \text{, otherwise} \end{cases}$$

•Construct approximations of this type on each subinterval
This method uses Lagrangian interpolants
•Endpoints are called *breakpoints*
•For higher polynomial degree we need more conditions
- e.g. specify values at points inside the interval $[x_k < x < x_{k+1}]$

•Specifying function and derivative values at the end points x_k, x_{k+1} leads to cubic Hermite interpolation

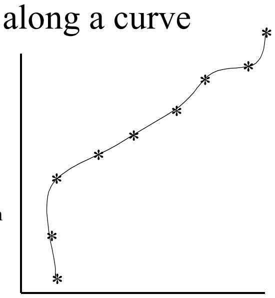
Cubic Spline

- Splines – name given to a flexible piece of wood used by draftsmen to draw curves through points.
  - Bend wood piece so that it passes through known points and draw a line through it.
  - Most commonly used interpolant used is the cubic spline
  - Provides continuity of the function, 1st and 2nd derivatives at the breakpoints.
  - Given n+1 points we have n intervals $\{x_i, f_i\}, \; i = 1, \dots, n+1$
  - Each polynomial has four unknown coefficients $P_i(x) = f_i \quad i = 1, \cdots, n+1$
    - Specifying function values provides 2 equations $P''_{i-1}(x) = P''_i(x) \quad i = 2, \cdots, n$
    - Two derivative continuity equations provides two more $P'_{i-1}(x) = P'_i(x) \quad i = 2, \cdots, n$

•Left with two free conditions. Usually chosen so that second derivatives are zero at ends

Interpolating along a curve

- Curve can be given as *x(s)* and *y(s)*
- *Given* x_i, y_i, s_i
- Can fit splines for x and y
- Can compute tangents, curvature and normal based on this fit
- Things like intensity van vary along the curve. Can also fit I(s)



Two and more dimensions

- Gradient $\nabla f = \frac{\partial f}{\partial x}\mathbf{e}_1 + \frac{\partial f}{\partial y}\mathbf{e}_2 = \frac{\partial f}{\partial x_i}\mathbf{e}_i$
- Directional derivative in the direction of a vector $\mathbf{n}$ $\nabla f \bullet \mathbf{n} = \frac{\partial f}{\partial x}\mathbf{e}_1 \bullet \mathbf{n} + \frac{\partial f}{\partial y}\mathbf{e}_2 \bullet \mathbf{n} = \frac{\partial f}{\partial x_i} n_i$

•Geometric interpretation
–∇f is normal to the surface $f(\mathbf{x}) = c$
– $\mathbf{n} = \nabla f / |\nabla f|$
•Taylor series

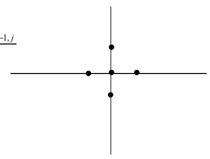
$$f(\mathbf{x} + \mathbf{h}) = f(\mathbf{x}) + \mathbf{h} \bullet \nabla f(\mathbf{x}) + \frac{1}{2}(\mathbf{h}\mathbf{h}) : \nabla\nabla f(\mathbf{x}) + O(|\mathbf{h}|^3)$$

$$f(\mathbf{x} + \mathbf{h}) = f(\mathbf{x}) + h_i \frac{\partial f}{\partial x_i} + \frac{1}{2} h_i h_j \frac{\partial}{\partial x_i}\frac{\partial f}{\partial x_j} + O(|\mathbf{h}|^3)$$

Finite differences

- Follows a similar pattern. One dimensional partial derivatives are calculated the same way.
- Multiple dimensional operators are computed using multidimensional stencils.

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} = \frac{f_{i+1,j} + f_{i,j+1} - 4f_{i,j} + f_{i,j-1} + f_{i-1,j}}{h^2}$$



Interpolation

- Polynomial interpolation in multiple dimensions
- Pascals triangle
- Least squares
- Move to a local coordinate system

Tensor product splines

- Splines form a local basis.
- Take products of one dimensional basis functions to make a basis in the higher dimension.

NURBS

- Used for precisely specifying n-d data.
- October 3 Tapas Kanungo, NURBS: Non-Uniform Rational B-Splines

Derivative of a matrix

Suppose $f(\mathbf{x})$ is a scalar-valued function of d variables x_i, $i = 1, 2, ...d$, which we represent as the vector $\mathbf{x}$. Then the derivative or gradient of f with respect to this vector is computed component by component, i.e.,

$$\nabla f(\mathbf{x}) = \mathrm{grad} f(\mathbf{x}) = \frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} = \begin{pmatrix} \frac{\partial f(\mathbf{x})}{\partial x_1} \\ \frac{\partial f(\mathbf{x})}{\partial x_2} \\ \vdots \\ \frac{\partial f(\mathbf{x})}{\partial x_d} \end{pmatrix}. \qquad (12)$$

If we have an n-dimensional vector-valued function $\mathbf{f}$ (note the use of boldface), of a d-dimensional vector $\mathbf{x}$, we calculate the derivatives and represent them as the *Jacobian matrix*

$$\mathbf{J}(\mathbf{x}) = \frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}} = \begin{pmatrix} \frac{\partial f_1(\mathbf{x})}{\partial x_1} & \cdots & \frac{\partial f_1(\mathbf{x})}{\partial x_d} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n(\mathbf{x})}{\partial x_1} & \cdots & \frac{\partial f_n(\mathbf{x})}{\partial x_d} \end{pmatrix}. \qquad (13)$$

If this matrix is square, its determinant (Sect. A.2.5) is called simply the *Jacobian* or occassionally the *Jacobian determinant.*

Jacobian and Hessian

We first recall the use of second derivatives of a scalar function of a scalar x in writing a Taylor series (or Taylor expansion) about a point:

$$f(x) = f(x_0) + \left.\frac{df(x)}{dx}\right|_{x=x_0} (x - x_0) + \frac{1}{2!} \left.\frac{d^2 f(x)}{dx^2}\right|_{x=x_0} (x - x_0)^2 + O((x - x_0)^3). \qquad (20)$$

Analogously, if our scalar-valued f is a instead function of a vector $\mathbf{x}$, we can expand $f(\mathbf{x})$ in a Taylor series around a point $\mathbf{x}_0$:

$$f(\mathbf{x}) = f(\mathbf{x}_0) + \left[\underbrace{\frac{\partial f}{\partial \mathbf{x}}}_{\mathbf{J}}\right]^t_{\mathbf{x}=\mathbf{x}_0} (\mathbf{x} - \mathbf{x}_0) + \frac{1}{2!} (\mathbf{x} - \mathbf{x}_0)^t \left[\underbrace{\frac{\partial^2 f}{\partial \mathbf{x}^2}}_{\mathbf{H}}\right]^t_{\mathbf{x}=\mathbf{x}_0} (\mathbf{x} - \mathbf{x}_0) + O(||\mathbf{x} - \mathbf{x}_0||^3), \quad (21)$$

where $\mathbf{H}$ is the *Hessian* matrix, the matrix of second-order derivatives of $f(\cdot)$, here