$\nabla \phi(\vec{x}) = 0 = \vec{f}(\vec{x})$ Same problem we already solved

recall old solution $\vec{\vec{J}}(\vec{x}^{(k-1)})(\vec{x}^{(k)} - \vec{x}^{(k-1)}) = -\vec{f}(\vec{x})^{(k-1)}$

$$\Rightarrow \quad J_{\nabla\phi}(\vec{x}^{(k-1)}) * (\vec{x}^{(k)} - \vec{x}^{(k-1)}) = -\nabla\phi(\vec{x})^{(k-1)}$$

where $J_{\nabla\phi}(\vec{x}) = H(\vec{x}) \equiv$ Hessian

$$H(\vec{x}) = \left\{ \frac{\partial \phi}{\partial x_i \partial x_j} \right\} = \begin{bmatrix} \frac{\partial^2\phi}{\partial x_0^2} & \cdots & \frac{\partial^2\phi}{\partial x_0 \partial x_{n-1}} \\ \vdots & & \\ \frac{\partial^2\phi}{\partial x_{n-1}\partial x_0} & \cdots & \frac{\partial^2\phi}{\partial x_{n-1}^2} \end{bmatrix}$$

Action

Single particle in one-dim (x)

$KE = \frac{1}{2} m \dot{x}^2 \qquad \dot{x} = dx/dt = v$

$V = V(x(t))$ potential

Lagrangian $\mathcal{L}(x(t), \dot{x}(t)) = KE(\dot{x}(t)) - V(x(t))$

We study particle from $t=0$ to $t=\mathcal{T}$ and define the action functional

$$S[x(t)] = \int_0^{\mathcal{T}} dt\, \mathcal{L}(x(t), \dot{x}(t))$$

S is a functional $\Rightarrow$ function of a function

$f(x) \longrightarrow S \longrightarrow \#$ depends on all values of x
but not of t

For given trajectory $x(t)$ from $t=0$ to $t=\tau$

$S[x(t)]$ gives a single number

Hamilton's principle: particle follows a trajectory
that minimizes the action

(Similar to Fermat's principle for path of least
time for light)

Algorithm: Discretize time $t_k = k\frac{\tau}{n-1} = kh_t$

where $k = 0, 1, \ldots n-1$ Let $x_k = x(t_k)$

We will compute an integral by a rectangle approx
and use forward diff. for $\dot{x}$

$$S_n = \sum_{k=0}^{n-2} \underset{\text{rect width}}{h_t} \underbrace{\left[\frac{1}{2} m \left(\frac{x_{k+1} - x_k}{h_t} \right)^2 - V(x_k) \right]}_{\text{rect height}}$$

we keep $x(0)$ and $x(\tau)$ fixed by definition

$S_n = S_n(x_1, x_2, \ldots x_{n-2}) \leftarrow$ need its minimum
like $\phi(\vec{x})$ problem

Solution

$$J_{\nabla\phi}(\vec{x}^{(k-1)})(\vec{x}^{(k)} - \vec{x}^{(k-1)}) = -\nabla\phi(\vec{x}^{(k-1)})$$

where we use S instead of ϕ

$$\nabla S_i = \frac{\partial S_n}{\partial x_i} = \sum_{k=0}^{n-2} h_t \left[\frac{m}{h_t^2} (x_{k+1} - x_k)(\delta_{i,k+1} - \delta_{i,k}) - \frac{\partial V(x_k)}{\partial x_i} \delta_{i,k} \right]$$

$$= \frac{m}{h_t}(2x_i - x_{i-1} - x_{i+1}) - h_t \frac{\partial V(x_i)}{\partial x_i}$$

also need $J_{\nabla S}$ $\Rightarrow$ take another derivative

(Eqn 5.137)

Approximation

We have data at some (unequally spaced) points

position X_j at time t_j $j = 0, 1, \ldots n-1$

We want $v_j = \frac{dx_j}{dt_j}$ at other locations

Need approx function $p(t)$

Problems

approximate $f(x)$ given selected $f(x_j)$ values

typically choose n basis functions $\phi_k(x)$

with a set of n unknown params c_k $k = 0, 1, \ldots n-1$

linear form $\Rightarrow p(x) = \sum_{k=0}^{n-1} c_k \phi_k(x)$

$\phi_k(x)$ do not have to be linear functions though

$p(x)$ and $\phi_k(x)$ valid for all x (not just x_j)

Interpolation: given (x_j, y_j) for $j = 0, 1, \ldots n-1$

use $\phi_k(x)$ to determin n params of c_k

n unknowns and n data points

$\Rightarrow$ we enforce that $p(x)$ go through exactly

all the input data points

$$p(x_j) = y_j = \sum_k c_k \phi_k(x_j)$$

$$\begin{pmatrix} \phi_0(x_0) & \phi_1(x_0) & \cdots & \phi_{n-1}(x_0) \\ \phi_0(x_1) & & & \\ \vdots & & & \\ \phi_0(x_{n-1}) & \phi_1(x_{n-1}) & \cdots & \phi_{n-1}(x_{n-1}) \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_{n-1} \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \end{pmatrix}$$

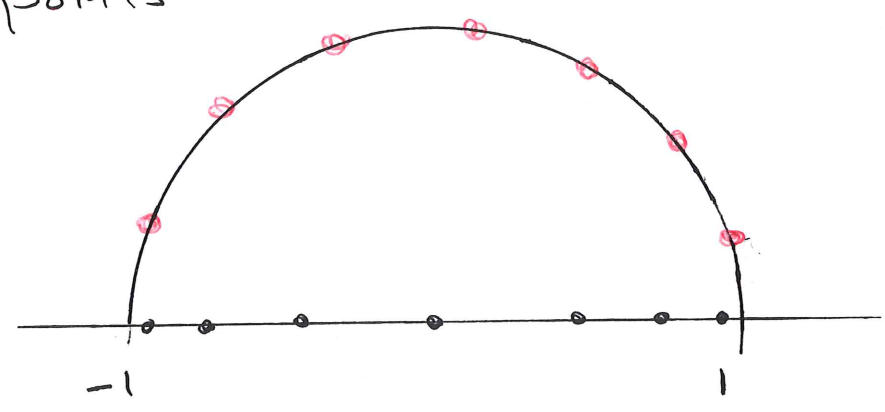
or $\Phi \vec{c} = \vec{y}$ $\quad \Phi$ $n \times n$, $\vec{c}$ column vector

Types of choices for $\phi(x)$: polynomial
poly (in intervals, piece-wise)
trigonometric

a single polynomial can do a pretty good job at approximating a complex function
→ helps to pick x_j a certain way
(called interpolation nodes)

Chebyshev nodes
$$x_j = -\cos\left(j \frac{\pi}{n-1}\right) \quad j = 0, 1, \ldots n-1$$

projection along x-axis of equidistant points on unit circle $\quad x_j \in [-1, 1]$



points cluster ends of interval

poly basis functions: $\phi_k(x) = x^k$

$$\{ 1, x, x^2, \dots x^{n-1} \}$$

$$p(x) = c_0 + c_1 x + c_2 x^2 + c_3 x^3 + \dots c_{n-1} x^{n-1}$$

Evaluate at our x_j $\quad p(x_j) = y_j = c_0 + c_1 x_j + \dots c_{n-1} x_j^{n-1}$

and solve for c params

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^{n-1} \\ 1 & x_0 & x_1^2 & & \\ \vdots & \vdots & \vdots & & \\ 1 & x_0 & x_1^2 & \cdots & x_{n-1}^{n-1} \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_{n-1} \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \end{pmatrix}$$

non-zero determinant for distinct x_j

$\rightarrow$ unique solution

Consider $\quad f(x) = \dfrac{1}{1+25x^2} \quad$ Choose $n = 101$

find solution [code]

Some issues for large n, due to monomial basis

Other poly basis choices do fine for same fcn

See Lagrange interpolation

Lagrange polynomials

$$L_k(x) = \frac{\prod_{j=0, j\neq k}^{n-1} (x - x_j)}{\prod_{j=0, j\neq k}^{n-1} (x_k - x_j)} \qquad k = 0, 1, \ldots n-1$$

denom depends only on interpolation points x_j's

num is polynomial in x, degree $n-1$

$L_k(x)$ has zeroes at x_j when $j \neq k$

$n = 3$ $\Downarrow$ quadratic

$$L_0(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} \qquad L_1(x) = \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)}$$

$$L_2(x) = \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}$$

$L_0(x_1) = L_0(x_2) = 0, \quad L_0(x_0) = 1$

$$L_k(x_j) = \delta_{kj} \qquad \delta_{kj} = \begin{cases} 1 & k = j \\ 0 & k \neq j \end{cases}$$

We use $L_k(x)$ as our basis functions

$$p(x) = \sum_{k=0}^{n-1} c_k \phi_k(x) = \sum_{k=0}^{n-1} c_k L_k(x)$$

but we require $p(x_j) = y_j$ by definition

$$p(x_j) = \sum_{k=0}^{n-1} c_k L_k(x_j) = \sum c_k \delta_{kj} = c_j = y_j$$