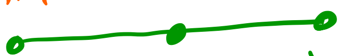
Long Stencil and Compact Discretizations

until now: 3-pt stencil

$$\frac{d}{dx}: \underbrace{\underbrace{D_+, D_-}_{2pt}, \tilde{D}}_{3pt}$$



3 degree of freedom

$$a u_{i-1} + b u_i + c u_{i+1}$$

$$= C_0 u_i + C_1 u_i' + C_2 u_i'' + \boxed{\text{Error term}}$$

$$\frac{d}{dx} = \tilde{D} + \boxed{O(h^2)}$$

$$\boxed{\tilde{D} u_i = \frac{u_{i+1} - u_{i-1}}{2h}}$$

d/dx

$C_0 = 0$

$C_1 = 1$

$C_2 = 0$

$$\frac{d^2}{dx^2} = \boxed{D^2} + \boxed{O(h^2)}$$

$$D^2 u_i = \frac{u_{i-1} - 2u_i + u_{i+1}}{h^2}$$

$C_0 = 0$, $C_1 = 0$, $C_2 = 1$

Long-Stencil

5-pt stencil

$i-2 \quad i-1 \quad i \quad i+1 \quad i+2$

***

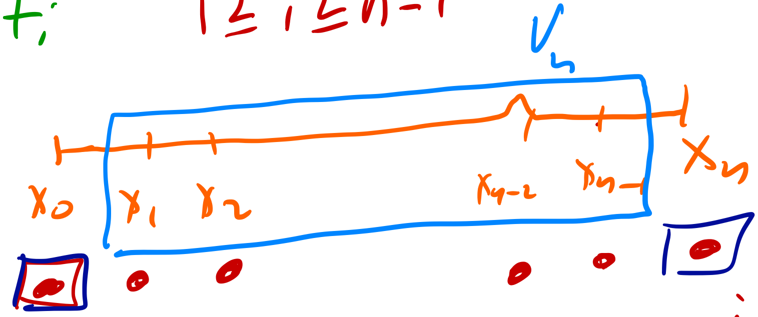
$$\boxed{\frac{d}{dx}u(x_i) = \frac{u_{i-2} - 8u_{i-1} + 0 \cdot u_i + 8u_{i+1} - u_{i+2}}{12} + O(h^4)}$$

Let D^2_{Long} = long stencil approx of $\frac{d^2}{dx^2}$ to $O(h^4)$

$$= \frac{c_{-2}u_{i-2} + c_{-1}u_{i-1} + c_0 u_i + c_1 u_{i+1} + c_2 u_{i+2}}{c(h)}$$

$$\begin{cases} \boxed{u'' = F} \\ \boxed{u(0) = u(1) = 0} \\ \boxed{u_0} \quad \boxed{u_n} \end{cases}$$

$D^2 u_i = F_i$ $\qquad 1 \le i \le n-1$



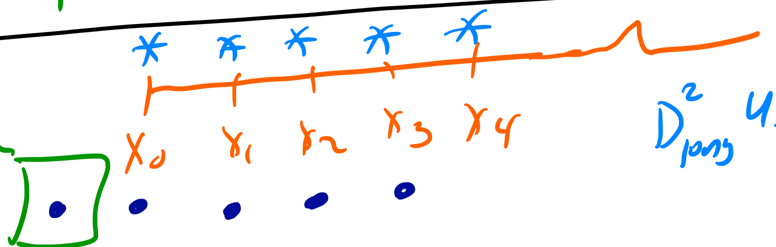
$$D^2 u_1 = \frac{u_0 - 2u_1 - u_2}{h^2}$$

$i=1$:

$i = n-1$

$$D^2 u_i = \frac{u_{i-1} - 2u_i + u_{i+1}}{h}$$

undefined!



$D^2_{long} u_2$ has no issue

$$D^2_{long} u_1 = \frac{\boxed{c_{-2} u_{-1}} + c_{-1} u_0 + c_0 u_1 + c_1 u_2 + c_2 u_3}{c(h)}$$

Issue: Long stencils give you ① Higher Order

However, issues arise ② at the boundary

Alternate Approach: Compact Operators

$$\frac{d^2}{dx^2}u(x_i) = \frac{u_{i-1} - 2u_i + u_{i+1}}{h^2} + O(h^2)$$

$$= D^2 + O(h^2) \Rightarrow \frac{d^2}{dx^2} = D^2 + O(h^2)$$

A more careful use of Taylor series give

$$\boxed{D^2} u_i = \frac{d^2}{dx^2}u(x_i) + \frac{2h^2}{4!}u_i^{(4)} + \frac{2h^4}{6!}u_i^{(6)} + \frac{2h^6}{8!}u_i^{(8)} + \dots$$

$$= \boxed{\frac{d^2}{dx^2}} u(x_i) + \boxed{\frac{h^2}{12}u_i^{(4)}} + \boxed{O(h^4)}$$

$$h^4 \boxed{\left(\frac{2h^2}{8!} \dots\right)}$$

or $\boxed{D^2} = \frac{d^2}{dx^2} + \frac{h^2}{12}\frac{d^4}{dx^4} + O(h^4)$

$$= \frac{d^2}{dx^2}\left(1 + \frac{h^2}{12}\boxed{\frac{d^2}{dx^2}}\right) + O(h^4)$$

using $\frac{d^2}{dx^2} = D^2 + O(h^2)$

$$= \frac{d^2}{dx^2}\left(1 + \frac{h^2}{12}\left(\boxed{D^2 + O(h^2)}\right)\right) + O(h^4)$$

$$= \frac{d^2}{dx^2}\left(1 + \frac{h^2}{12}D^2 + O(h^4)\right) + O(h^4)$$

$$= \boxed{\frac{d^2}{dx^2}\left(1 + \frac{h^2}{12}D^2\right) + O(h^4)}$$

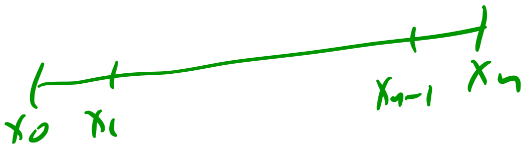
$$\Rightarrow \quad D^2 = \frac{d^2}{dx^2}\boxed{\left(1+\frac{h^2}{12}D^2\right)} + O(h^4)$$

$$\boxed{\frac{D^2}{1+\frac{h^2}{12}D^2}} = \frac{d^2}{dx^2} + \boxed{\frac{O(h^4)}{\left(1+\frac{h^2}{12}D^2\right)}} \qquad O(h^4)$$

$$\boxed{\frac{d^2}{dx^2} = \boxed{\frac{D^2}{1+\frac{h^2}{12}D^2}} + O(h^4)}$$

Compact Operator

In practice:



2nd order

3-pt stencil

$$D^2 u_i = F_i \quad \Rightarrow \quad A_h V_h = F_h$$

$$1 \le i \le n-1 \qquad (n-1)\times(n-1)$$

Linear system

4th order (Dirichlet BCs)

?

$$\frac{D^2}{1+\frac{h^2}{12}D^2} u_i = F_i \qquad 1 \le i \le n-1$$

$$= R(D^2) \qquad R(x) = \frac{x}{1+\frac{h^2}{12}x} \quad \text{rational function}$$

one way to apply this is

$$D^2 u_i = \boxed{\left(1+\frac{h^2}{12}D^2\right)F_i} \quad \text{pre-processing}$$

$$1 \le i \le n-1$$

$\left(1+\frac{h^2}{12}\underline{D^2}\right)\underline{F_1}$ involves $\underline{x_0}, x_1, x_2$

4^{th} order Dirichlet Solver

```
function [vh,xh] = poiss1d_4P(f,N)
%
%  4th order 1D Poisson solve with DST
%  implemented by matrix multiply by P
%
%  u'' = f,  u(0)=u(1)=0
%

h = 1/N;
xh = (0:N)'*h;                         % discrete grid
fh = f(xh);
lam = -2*(1-cos(xh(2:N)*pi))/(h^2);  % evals
P = createP(N);

fh = fh(2:N)+(1/12)*(fh(1:N-1)-2*fh(2:N)+fh(3:N+1));
vh = P*fh;     % transform
vh = vh./(lam);   % Solve in Fourier space
vh = P*vh;        % transform back
vh = [0; vh; 0];  % set BC
```

$\left(1+\frac{h^2}{12}D^2\right)\cdot F_i$

$1 \le i \le n-1$

$(\text{or } 2:N)$

2nd Order Neumann Solver

```
function [vh,xh] = poiss1d_2PNeumann(N)
%  2nd order Poisson solve using P/DCT
%  with homogeneous Neumann BC
%
%  u'' = f, u'(0)=u'(1)=0
%
%  u(x) = exp(cos(pi*x))
%  f(x) = (pi^2)*exp(cos(pi*x)).*(sin(pi*x).^2-cos(pi*x))

h = 1/N;  xh = (0:N)'*h;
lam = -2*(1-cos(xh(1:N+1)*pi))/(h^2);
f =  pi^2*(-cos(pi*xh)+sin(pi*xh).^2).*exp(cos(pi*xh));
P = createcosP(N);

vh = P*f;                                   % transform
vh(1) = 0;                                  % set 0 mode
vh(2:N+1) = vh(2:N+1)./lam(2:N+1);
vh = P*vh;                                  % transform back

vh = vh + (exp(1)-vh(1));
err = max(abs(vh - exp(cos(pi*xh))))
```