

Title: Numerical Methods Lecture - 230131

Speakers: Erik Schnetter

Collection: Numerical Methods (2022/2023)

Date: January 31, 2023 - 9:15 AM

URL: <https://pirsa.org/23010008>

A screenshot of a Mac OS X desktop environment. At the top, a green Safari browser window is open, displaying the URL `symmetry.pi.local`. The title bar of the window shows "JupyterLab". The main content area of the browser displays a Jupyter Notebook cell titled "Untitled8.ipynb". The cell contains the following code and output:

```
[1]: versioninfo()
Julia Version 1.8.5
Commit 17cfb8e65ea (2023-01-08 06:45 UTC)
Platform Info:
  OS: Linux (x86_64-linux-gnu)
  CPU: 80 x Intel(R) Xeon(R) Gold 6148 CPU @ 2.40GHz
  WORD_SIZE: 64
  LIBM: libopenlibm
  LLVM: libLLVM-13.0.1 (ORCJIT, skylake-avx512)
  Threads: 1 on 80 virtual cores
Environment:
  LD_LIBRARY_PATH = /cm/shared/apps/slurm/19.05.8/lib64/slurm:/cm/shared/apps/slurm/19.05.8/lib64
  LD_LIBRARY_PATH_modshare = /cm/shared/apps/slurm/19.05.8/lib64:1:/cm/shared/apps/slurm/19.05.8/lib64/slurm:1
```

The bottom status bar of the browser window shows "Simple" mode, "0 \$ 1 ⚙️", "Julia 1.8.5 | Idle", "Mem: 451.46 MB", "Saving completed", "Mode: Command", "Ln 1, Col 14", and "Untitled8.ipynb". The right side of the screen shows the Mac OS X Dock with various application icons.

Safari File Edit View History Bookmarks Develop Window Help

symmetry.pi.local

JupyterLab

File Edit View Run Kernel Git Tabs Settings Help Mem:453 MB

Untitled8.ipynb

Code git Julia 1.8.5

```
[1]: versioninfo()

Julia Version 1.8.5
Commit 17cfb8e65ea (2023-01-08 06:45 UTC)
Platform Info:
  OS: Linux (x86_64-linux-gnu)
  CPU: 80 x Intel(R) Xeon(R) Gold 6148 CPU @ 2.40GHz
  WORD_SIZE: 64
  LIBM: libopenlibm
  LLVM: libLLVM-13.0.1 (ORCJIT, skylake-avx512)
  Threads: 1 on 80 virtual cores
Environment:
  LD_LIBRARY_PATH = /cm/shared/apps/slurm/19.05.8/lib64/slurm:/cm/shared/apps/slurm/19.05.8/lib64
  LD_LIBRARY_PATH_modshare = /cm/shared/apps/slurm/19.05.8/lib64:1:/cm/shared/apps/slurm/19.05.8/lib64/slurm:1
```

ODE

Equation: Solve harmonic oscillator

```
[ ]: # ḡ = -x
```

Simple 0 \$ 1 Julia 1.8.5 | Idle Mem: 453.04 MB Saving completed Mode: Edit Ln 1, Col 10 Untitled8.ipynb

Safari File Edit View History Bookmarks Develop Window Help symmetry.pi.local JupyterLab Tue Jan 31 09:31

File Edit View Run Kernel Git Tabs Settings Help Mem:519 MB

Untitled8.ipynb Julia 1.8.5

```
# X = -X
```

```
[3]: function f(y)
    x, v = y
    ẋ = v
    v̇ = -x
    ẏ = [ẋ, v̇]
    return ẏ
end
```

```
[3]: f (generic function with 1 method)
```

```
[4]: function euler(f, y₀, h)
    k₀ = f(y₀)
    y₁ = y₀ + h * k₀
    return y₁
end
```

```
[4]: euler (generic function with 1 method)
```

```
[ ]:
```

Simple 0 \$ 1 Julia 1.8.5 | Idle Mem: 519.49 MB Saving completed Mode: Edit Ln 5, Col 4 Untitled8.ipynb

Safari File Edit View History Bookmarks Develop Window Help

symmetry.pi.local JupyterLab

File Edit View Run Kernel Git Tabs Settings Help Mem:591 MB

Untitled8.ipynb

Code git Julia 1.8.5

```
[4]: function euler(t, y₀, n)
    k₀ = f(y₀)
    y₁ = y₀ + h * k₀
    return y₁
end

[4]: euler (generic function with 1 method)

[5]: y₀ = [1.0, 0.0]

[5]: 2-element Vector{Float64}:
     1.0
     0.0

[7]: euler(harmonic, y₀, 0.1)

[7]: 2-element Vector{Float64}:
     1.0
    -0.1

[ ]: function evolve(f, y, h, nsteps)
    for n in 1:nsteps
        y = euler(f, y, h)
    end
end
```

Simple 0 \$ 1 Julia 1.8.5 | Idle Mem: 590.56 MB Saving completed Mode: Edit Ln 1, Col 33 Untitled8.ipynb

Safari File Edit View History Bookmarks Develop Window Help symmetry.pi.local JupyterLab Tue Jan 31 09:36

File Edit View Run Kernel Git Tabs Settings Help Mem:925 MB

Untitled8.ipynb Julia 1.8.5

```
end

[8]: evolve (generic function with 1 method)

[10]: ys = evolve(harmonic, y₀, 0.1, 100);

[11]: using WGLMakie

[*]: fig = Figure()
       ax = Axis(fig[1, 1])
       plot!(map(y -> y[1], ys), map(y -> y[2]))
       fig

[ ]:
```

Simple 0 \$ 1 Julia 1.8.5 | Busy Mem: 924.76 MB Saving completed Mode: Edit Ln 1, Col 1 Untitled8.ipynb

Safari File Edit View History Bookmarks Develop Window Help symmetry.pi.local JupyterLab Tue Jan 31 09:35

File Edit View Run Kernel Git Tabs Settings Help Mem:1.18 GB

Untitled8.ipynb Julia 1.8.5

```
[7]: 2-element Vector{Float64}:
      1.0
     -0.1

[8]: function evolve(f, y, h, nsteps)
    trajectory = [y]
    for n in 1:nsteps
        y = euler(f, y, h)
        push!(trajectory, y)
    end
    return trajectory
end

[8]: evolve (generic function with 1 method)

[10]: ys = evolve(harmonic, y₀, 0.1, 100);

[11]: using WGLMakie

[ ]: fig = Figure()
ax = Axis(fig[1, 1])
plot!(map(y → y[1], ys), map(y → ))
```

Simple 0 \$ 1 Julia 1.8.5 | Idle Mem: 1.18 GB Saving completed Mode: Edit Ln 3, Col 35 Untitled8.ipynb

Safari File Edit View History Bookmarks Develop Window Help symmetry.pi.local JupyterLab Tue Jan 31 09:38

File Edit View Run Kernel Git Tabs Settings Help Mem:1.06 GB

Untitled8.ipynb Julia 1.8.5

```
[14]: ys = evolve(harmonic, y₀, 0.05, 200);
```

```
[15]: using WGLMakie
```

```
[16]: fig = Figure()
ax = Axis(fig[1, 1])
plot!(map(y -> y[1], ys), map(y -> y[2], ys))
fig
```

[16]:

Simple 0 \$ 1 Julia 1.8.5 | Idle Mem: 1.06 GB Saving completed Mode: Edit Ln 1, Col 29 Untitled8.ipynb

Safari File Edit View History Bookmarks Develop Window Help

symmetry.pi.local JupyterLab

File Edit View Run Kernel Git Tabs Settings Help Mem:1.19 GB

Untitled8.ipynb

[15]: `using WGLMakie`

[20]: `fig = Figure()
ax = Axis(fig[1, 1])
plot!(map(y -> y[1], ys), map(y -> y[2], ys); color=:red)
plot!(map(y -> y[1], ys2), map(y -> y[2], ys2); color=:green)
fig`

[20]:

Simple 0 \$ 1 Julia 1.8.5 | Idle Mem: 1.19 GB Saving completed Mode: Command Ln 1, Col 1 Untitled8.ipynb

Safari File Edit View History Bookmarks Develop Window Help symmetry.pi.local JupyterLab Tue Jan 31 09:43

File Edit View Run Kernel Git Tabs Settings Help Mem:1.21 GB

Untitled8.ipynb Julia 1.8.5

```
push!(trajectory, y)
end
return trajectory
end

[22]: evolve_midpoint (generic function with 1 method)

[23]: ys_m = evolve_midpoint(harmonic, y₀, 0.1, 100);

[24]: fig = Figure()
ax = Axis(fig[1, 1])
plot!(map(y -> y[1], ys), map(y -> y[2], ys); color=:red)
plot!(map(y -> y[1], ys₂), map(y -> y[2], ys₂); color=:green)
fig
```

[]:

Simple 0 \$ 1 Julia 1.8.5 | Idle Mem: 1.21 GB Saving completed Mode: Edit Ln 4, Col 25 Untitled8.ipynb

Safari File Edit View History Bookmarks Develop Window Help symmetry.pi.local JupyterLab Tue Jan 31 09:44

File Edit View Run Kernel Git Tabs Settings Help Mem:1.08 GB

Untitled8.ipynb × Julia 1.8.5

[29]:

```
plot!(map(y -> y^11, ys_III), map(y -> y^11, ys_III); color=:green)
fig
```

Simple

0 \$ 1 ⚙



Julia 1.8.5 | Idle

Mem: 1.08 GB

Saving completed

Mode: Command



Ln 1, Col 1

Untitled8.ipynb

Safari File Edit View History Bookmarks Develop Window Help symmetry.pi.local JupyterLab Mem:1.22 GB

File Edit View Run Kernel Git Tabs Settings Help

Untitled8.ipynb Julia 1.8.5

```
[33]: fig = figure()
ax = Axis(fig[1, 1])
plot!(map(y -> y[1], ys_m1), map(y -> y[2], ys_m1); color=:red)
plot!(map(y -> y[1], ys_m2), map(y -> y[2], ys_m2); color=:green)
fig
```

[33]:

Simple 0 \$ 1 Julia 1.8.5 | Idle Mem: 1.22 GB Saving completed Mode: Command Ln 1, Col 1 Untitled8.ipynb

Safari File Edit View History Bookmarks Develop Window Help symmetry.pi.local JupyterLab Tue Jan 31 09:47

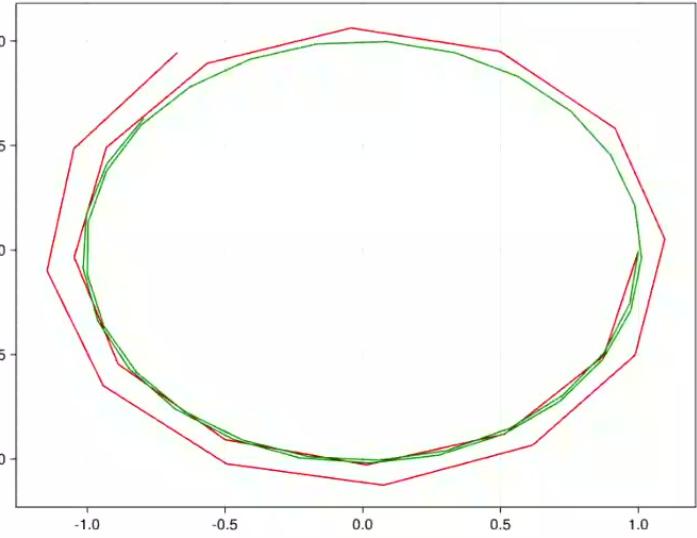
File Edit View Run Kernel Git Tabs Settings Help Mem:1.18 GB

Untitled8.ipynb × Julia 1.8.5

ys_m1 = evolve_imput(ym1, y0, 0.25, 40);

```
[34]: fig = Figure()
ax = Axis(fig[1, 1])
lines!(map(y -> y[1], ys_m1), map(y -> y[2], ys_m1); color=:red)
lines!(map(y -> y[1], ys_m2), map(y -> y[2], ys_m2); color=:green)
fig
```

[34] :



Simple 0 \$ 1 Julia 1.8.5 | Idle Mem: 1.18 GB Saving completed Mode: Command Ln 1, Col 1 Untitled8.ipynb

Safari File Edit View History Bookmarks Develop Window Help Charged 7.8KB/s 1.0KB/s Tue Jan 31 09:55

JupyterLab docs.sciml.ai DifferentialEquations.jl: Efficient Differential Equation Solving in Julia - DifferentialEquations.jl

HOME MODELING ▾ SOLVERS ▾ ANALYSIS ▾ MACHINE LEARNING ▾ DEVELOPER TOOLS ▾

DifferentialEquations.jl: Efficient Differential Equation Solving in Julia... [Edit on GitHub](#) [⚙️](#)

DifferentialEquations.jl: Efficient Differential Equation Solving in Julia

This is a suite for numerically solving differential equations written in Julia and available for use in Julia, Python, and R. The purpose of this package is to supply efficient Julia implementations of solvers for various differential equations. Equations within the realm of this package include:

- Discrete equations (function maps, discrete stochastic (Gillespie/Markov) simulations)
- Ordinary differential equations (ODEs)
- Split and Partitioned ODEs (Symplectic integrators, IMEX Methods)
- Stochastic ordinary differential equations (SODEs or SDEs)
- Stochastic differential-algebraic equations (SDAEs)
- Random differential equations (RODEs or RDEs)
- Differential algebraic equations (DAEs)
- Delay differential equations (DDEs)
- Neutral, retarded, and algebraic delay differential equations (NDDEs, RDDEs, and DDAEs)

Search docs

DifferentialEquations.jl: Efficient Differential Equation Solving in Julia

- Contributing
- Supporting and Citing
- Getting Started: Installation And First Steps
- Acknowledgements

Version v7.7.0

Safari File Edit View History Bookmarks Develop Window Help

symmetry.pi.local

JupyterLab DifferentialEquations.jl: Efficient Differential Equation Solving in Julia - DifferentialEquations.jl

File Edit View Run Kernel Git Tabs Settings Help Mem:1.09 GB

Untitled8.ipynb

Code git Julia 1.8.5

```
plot!(map(y -> y[1], ys), map(y -> y[2], ys); color=:red)
plot!(map(y -> y[1], ys_m), map(y -> y[2], ys_m); color=:green)
fig
```

[29]:

[32]:

```
ys_m1 = evolve_midpoint(harmonic, y₀, 0.5, 20);
ys_m2 = evolve_midpoint(harmonic, y₀, 0.25, 40);
```

Simple 0 \$ 1 Julia 1.8.5 | Idle Mem: 1.09 GB Saving completed Mode: Command Ln 1, Col 1 Untitled8.ipynb

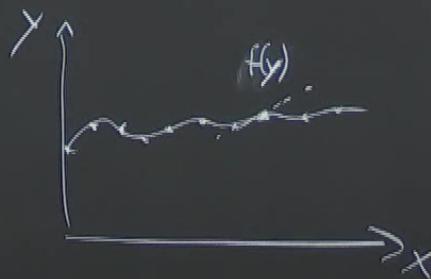
ODE

$$y'(x) = f(y, x)$$

$$\frac{y(x+h) - y(x)}{h} = f(y)$$

EULER

$$y(x+h) = y(x) + h \cdot f(y)$$

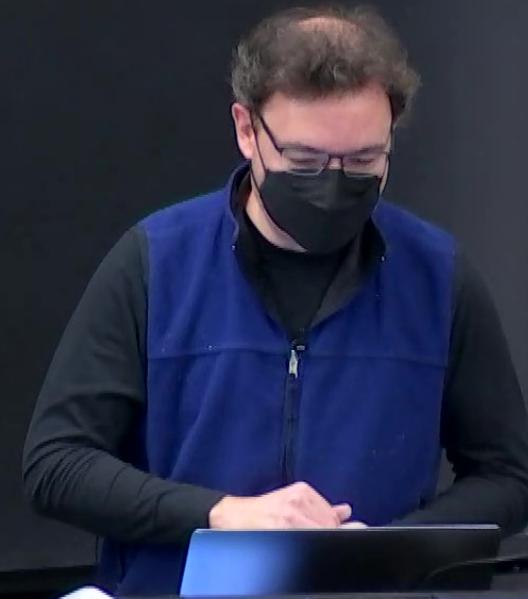


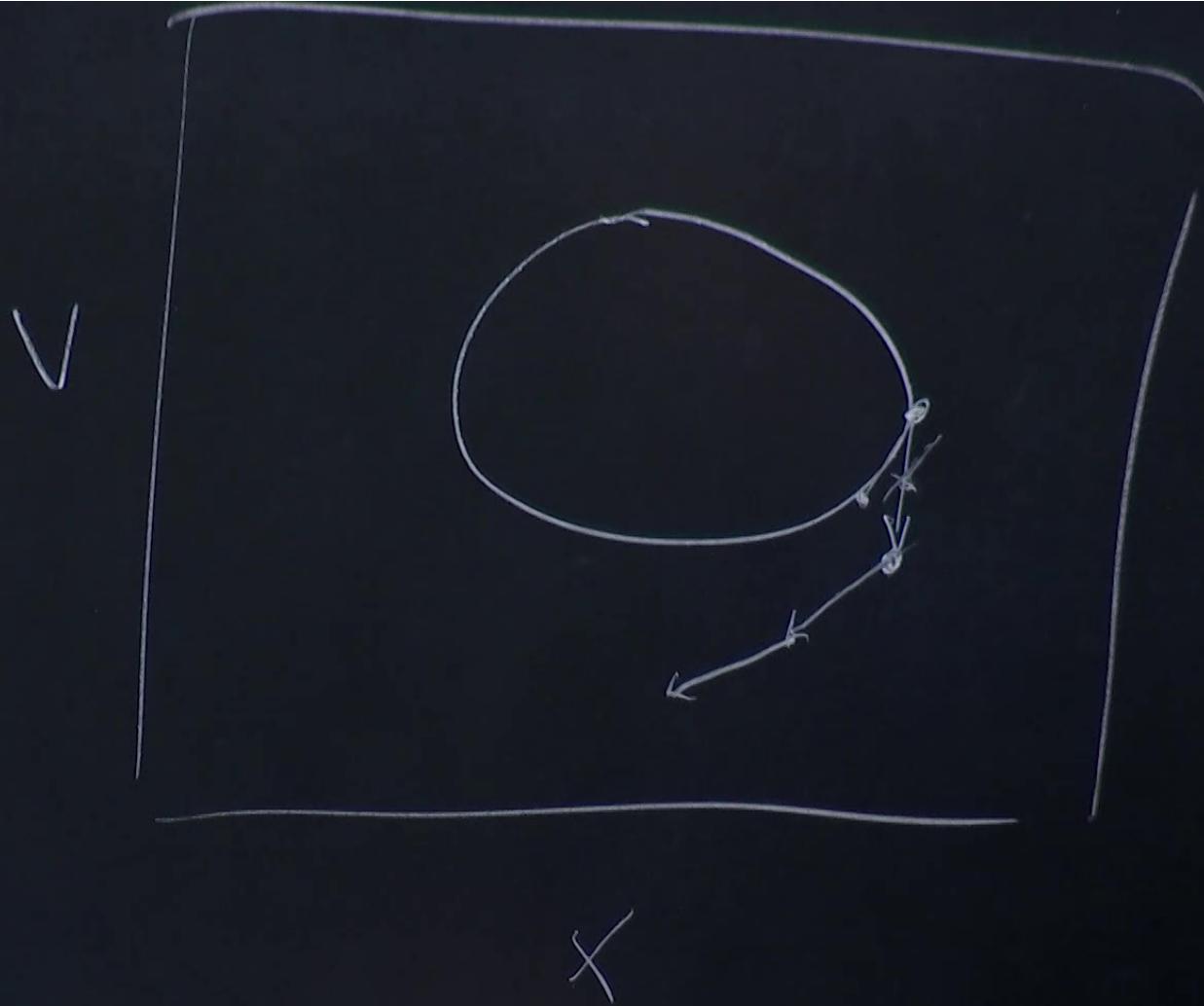
$$Y^0(x) = f(x)$$

$$\gamma_1 = \gamma$$

$$\gamma_2 = \gamma'$$

$$\boxed{\begin{array}{l} Y_1 = Y_2 \\ Y_1' = f(\gamma_1) \end{array}}$$





ODE

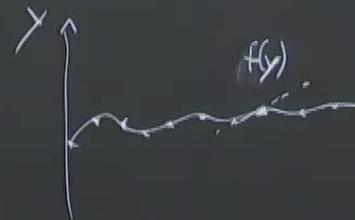
$$y'(x) = f(y, x)$$

$$\frac{y(x+h) - y(x)}{h} = f(y)$$

EULER

$$y(x+h) = y(x) + h \cdot f(y, x)$$

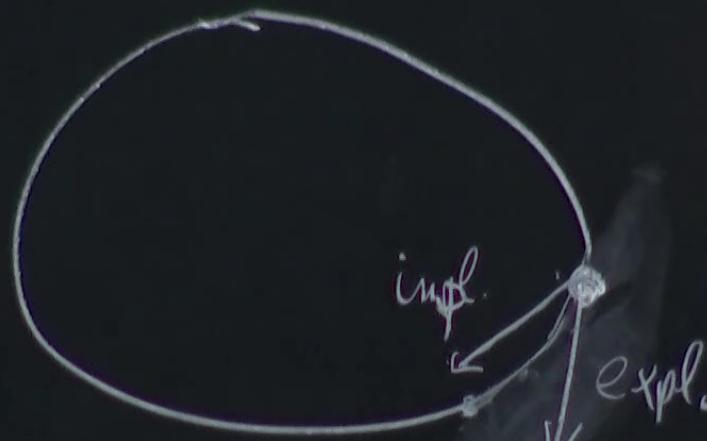
explicit



$$\frac{y(x) - y(x-h)}{h} = f(y, x)$$

$$y(x+h) = y(x) + h \cdot f(y, x+h)$$

implicit



Representation functions

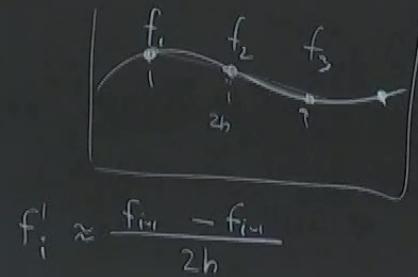
(pseudo-) spectral:

basis functions

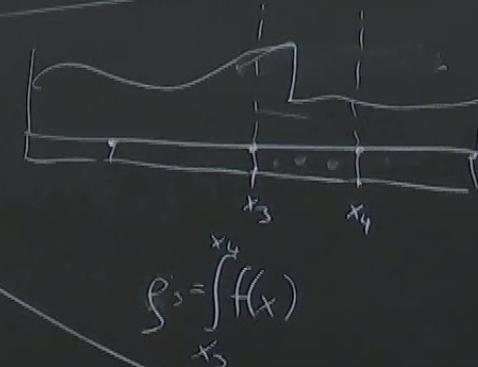
$$f(x) = \sum_i c^i b_i(x)$$

finite volume:

finite differences:



$$f'_i \approx \frac{f_{i+1} - f_{i-1}}{2h}$$



Convergence:



$$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$

Taylor: $f(x+h) = f(x) + h \cdot f'(x) + O(h^2)$

$$f'(x) = \frac{f(x+h) - f(x)}{h} + O(h)$$

$$f(x+h) = f(x) + h \cdot f'(x) + \frac{1}{2}h^2 f''(x) + O(h^3)$$

$$f(x-h) = f(x) - h \cdot f'(x) + \frac{1}{2}h^2 f''(x) + O(h^3)$$

Convergence order

P

$$\frac{f(x+h) - f(x-h)}{2h} + C(x) \cdot h^2 + O(h^4)$$

$$\frac{h \cdot f'(x) + O(h^2)}{(x) + O(h)}$$

$$f(x) + \frac{1}{2}h^2 f''(x) + O(h^3)$$

$$f'(x) + \frac{1}{2}h^2 f''(x) + O(h^3)$$

$$f_h(x) = f^*(x) + C(x) \cdot h^P + \dots$$

1. Convergence to known solution

$$h_1 \quad h_2$$

$$f_{h_1} = f^* + C h_1^P \quad \frac{f_1 - f^*}{f_2 - f^*} = \left(\frac{h_1}{h_2}\right)^P \rightarrow \text{solve for } P$$

$$f_{h_2} = f^* + C h_2^P$$

2. Self converge

$$h_1 \quad h_2 \quad h_3 \rightarrow P$$

3. Richardson extrapolation

$$\rightarrow \text{solve for } f^*$$

$$f^*(x) + C(x) \cdot h^P$$

known solution

$$+ C h_1^P$$

$$\frac{f_1 - f^*}{f_2 - f^*} = \left(\frac{h_1}{h_2}\right)^P \quad \rightarrow \text{solve for } P$$

$$+ C h_2^P$$

3. Richardson extrapolation

$$\rightarrow \text{solve for } f^*$$

$$h_1 = \alpha^2 h$$

$$h_2 = \alpha h$$

$$h_3 = h$$

$$f_{h_1} = f^* + C(\alpha^2 h)^P$$

$$f_{h_2} = f^* + C(\alpha h)^P$$

$$f_{h_3} = f^* + C h^P$$

$$\frac{f_{h_1} - f_{h_2}}{f_{h_2} - f_{h_3}} = \frac{(\alpha^2 h)^P - (\alpha h)^P}{(\alpha h)^P - h^P}$$

P

$$\left. \begin{array}{l}
 f_{h_1} = f^* + C (\alpha h)^P \\
 f_{h_2} = f^* + C (\alpha h)^P \\
 f_{h_3} = f^* + C h^P
 \end{array} \right\} \quad \text{solve for } P$$

$$\frac{f_{h_1} - f_{h_2}}{f_{h_2} - f_{h_3}} = \frac{(\alpha h)^P - (\alpha h)^P}{(\alpha h)^P - h^P} = \frac{(\alpha h)^P}{h^P} = \alpha$$

