Title: Solving initial-boundary value problems without numerical differentiation

Date: May 07, 2014 11:25 AM

URL: http://pirsa.org/14050042

Abstract: The numerical solution of nonlinear partial differential equations with nontrivial boundary conditions is central to many areas of modelling. When high accuracy is required (pseudo) spectral methods are usually the first choice. Typically in this approach we search for the pre-image under a linear operator which represents a combination of spatial derivatives along with the boundayr conditions in every time step. This operator can be quite ill-conditioned. On a basis of Chebyshev polynomials for instance the condition number increases algebraically with the number of basis functions. I will present an alternative method based on recent work by Viswanath and Tobasco which avoids numerical differentiation entirely through the use of Green's functions. I will demonstrate this method on the Kuramoto-Sivashinsky equation with fixed boundary conditions.

Crestron Electronics, Inc. 36"

Solving initial-boundary value problems without numerical differentiation

Lennaert van Veen

Outline

1. Motivation

Outline

1. Motivation
2. Using Green's function
3. Kuramoto-Sivashinsky

Outline

1. Motivation
2. Using Green's function
3. Kuramoto-Sivashinsky
4. Computing Green's function

Crestron Electronics, Inc. 36"

Outline

1. Motivation
2. Using Green's function
3. Kuramoto-Sivashinsky
4. Computing Green's function
5. Proper quadrature

Motivation – Setup

$u_t = Lu + f(u)$ e.g. Navier-Stokes, reaction-diffusion, ...

$$\left.\begin{array}{l} u(-1) = a_0,\ u'(-1) = a_1, \dots \\ u(1) = b_0,\ u'(1) = b_1, \dots \end{array}\right\} \text{sufficiently many BC}$$

$u(x,0) = u_0(x)$ initial value

Central question: what method enables us to find solutions

- with accurately resolved boundary layers;
- that are stable to overresolution;
- that are not prohibitively expensive to compute?

Step 1: implicit-explicit time discretization, e.g.

$$\frac{u^{k+1} - u^k}{\Delta} = Lu^{k+1} + f(u^k) \quad \text{or} \quad [\mathbb{I} - \Delta L]\, u^{k+1} = u^k + \Delta f(u^k)$$

- Balances stability with computation time.
- Gives linear BVP for u^{k+1} given u^k.
- Can be exended to higher order (SBDF).
- The properties of $\mathcal{L} = \mathbb{I} - \Delta L$ are quintessential.

Motivation – Setup

$$u_t = Lu + f(u)$$ e.g. Navier-Stokes, reaction-diffusion, ...

$$\left.\begin{array}{l} u(-1) = a_0,\ u'(-1) = a_1, \dots \\ u(1) = b_0,\ u'(1) = b_1, \dots \end{array}\right\}$$ sufficiently many BC

$$u(x,0) = u_0(x)$$ initial value

Central question: what method enables us to find solutions

- with accurately resolved boundary layers;
- that are stable to overresolution;
- that are not prohibitively expensive to compute?

Step 1: implicit-explicit time discretization, e.g.

$$\frac{u^{k+1} - u^k}{\Delta} = Lu^{k+1} + f(u^k) \quad \text{or} \quad [\mathbb{I} - \Delta L]\, u^{k+1} = u^k + \Delta f(u^k)$$

- Balances stability with computation time.
- Gives linear BVP for u^{k+1} given u^k.
- Can be exended to higher order (SBDF).
- The properties of $\mathcal{L} = \mathbb{I} - \Delta L$ are quintessential.

Step 1: implicit-explicit time discretization, e.g.

$$\frac{u^{k+1}-u^k}{\Delta} = Lu^{k+1} + f(u^k) \text{ or } [\mathbb{I} - \Delta L]\, u^{k+1} = u^k + \Delta f(u^k)$$

- Balances stability with computation time.
- Gives linear BVP for u^{k+1} given u^k.
- Can be exended to higher order (SBDF).
- The properties of $\mathcal{L} = \mathbb{I} - \Delta L$ are quintessential.

Step 2: solve the linear BVP

- by finite differences,
- by a (pseudo) spectral method,
- or using Green's function.

Step 1: implicit-explicit time discretization, e.g.

$$\frac{u^{k+1} - u^k}{\Delta} = Lu^{k+1} + f(u^k) \quad \text{or} \quad [\mathbb{I} - \Delta L]\, u^{k+1} = u^k + \Delta f(u^k)$$

- Balances stability with computation time.
- Gives linear BVP for u^{k+1} given u^k.
- Can be exended to higher order (SBDF).
- The properties of $\mathcal{L} = \mathbb{I} - \Delta L$ are quintessential.

Step 2: solve the linear BVP

- by finite differences,
- by a (pseudo) spectral method,
- or using Green's function.

Pros and cons:

- Finite differences: $u(x,t) \rightarrow \{u_j(t)\}$ where $u_j(t) \approx u(x_j,t)$.
  - Pro: $\mathcal{L}$ represented by a sparse, banded matrix so solving is fast.
  - Pro: Easy to program.
  - Con: Low accuracy – $O(h^q)$ for fixed q.
  - Con: Not stable to overresolution.

Pros and cons:

- Finite differences: $u(x,t) \rightarrow \{u_j(t)\}$ where $u_j(t) \approx u(x_j,t)$.
  - Pro: $\mathcal{L}$ represented by a sparse, banded matrix so solving is fast.
  - Pro: Easy to program.
  - Con: Low accuracy – $O(h^q)$ for fixed q.
  - Con: Not stable to overresolution.
- Pseudo spectral method: $u(x,t) \rightarrow \{a_k\}$ where $u(x,t) \approx \sum_k a_k(t) T_k(x)$.
  - Pro: High accuracy – "$q = \infty$".
  - Con: $\mathcal{L}$ represented by dense matrix.
  - Con: BC are *global* conditions on the variables.
  - Pro: For Chebyshev basis, $O(n \ln n)$ methods for computing $\mathcal{L}v$ available.
  - Con: Condition number of $\mathcal{L}$ + BC grows algebraically, e.g. $O(n^{2p})$ for Chebysev.

Green's function – *invert first*

Both ideas are based on *discretizing first, then inverting*. With Green's function, we can *invert exactly, then discretize*:

$$u^{(k+1)} = \int_{y=-1}^{1} G(x,y)\left(u^k(y) + \Delta f(u^k(y))\right) \mathrm{d}y$$

where $G \in C^{p-2}$ satisfies

$$\mathcal{L}G(x,y) = \Delta(x-y)$$

The RHS now requires *numerical integration* instead of *differentiation*, which is much more stable.
The question is: what quadrature to use?

Step 1: implicit-explicit time discretization, e.g.

$$\frac{u^{k+1} - u^k}{\Delta} = Lu^{k+1} + f(u^k) \quad \text{or} \quad [\mathbb{I} - \Delta L]\, u^{k+1} = u^k + \Delta f(u^k)$$

- Balances stability with computation time.
- Gives linear BVP for u^{k+1} given u^k.
- Can be exended to higher order (SBDF).
- The properties of $\mathcal{L} = \mathbb{I} - \Delta L$ are quintessential.

Green's function – *invert first*

Both ideas are based on *discretizing first, then inverting*. With Green's function, we can *invert exactly, then discretize*:

$$u^{(k+1)} = \int_{y=-1}^{1} G(x,y) \left(u^k(y) + \Delta f(u^k(y)) \right) \mathrm{d}y$$

where $G \in C^{p-2}$ satisfies

$$\mathcal{L} G(x,y) = \Delta(x-y)$$

The RHS now requires *numerical integration* instead of *differentiation*, which is much more stable.
The question is: what quadrature to use?

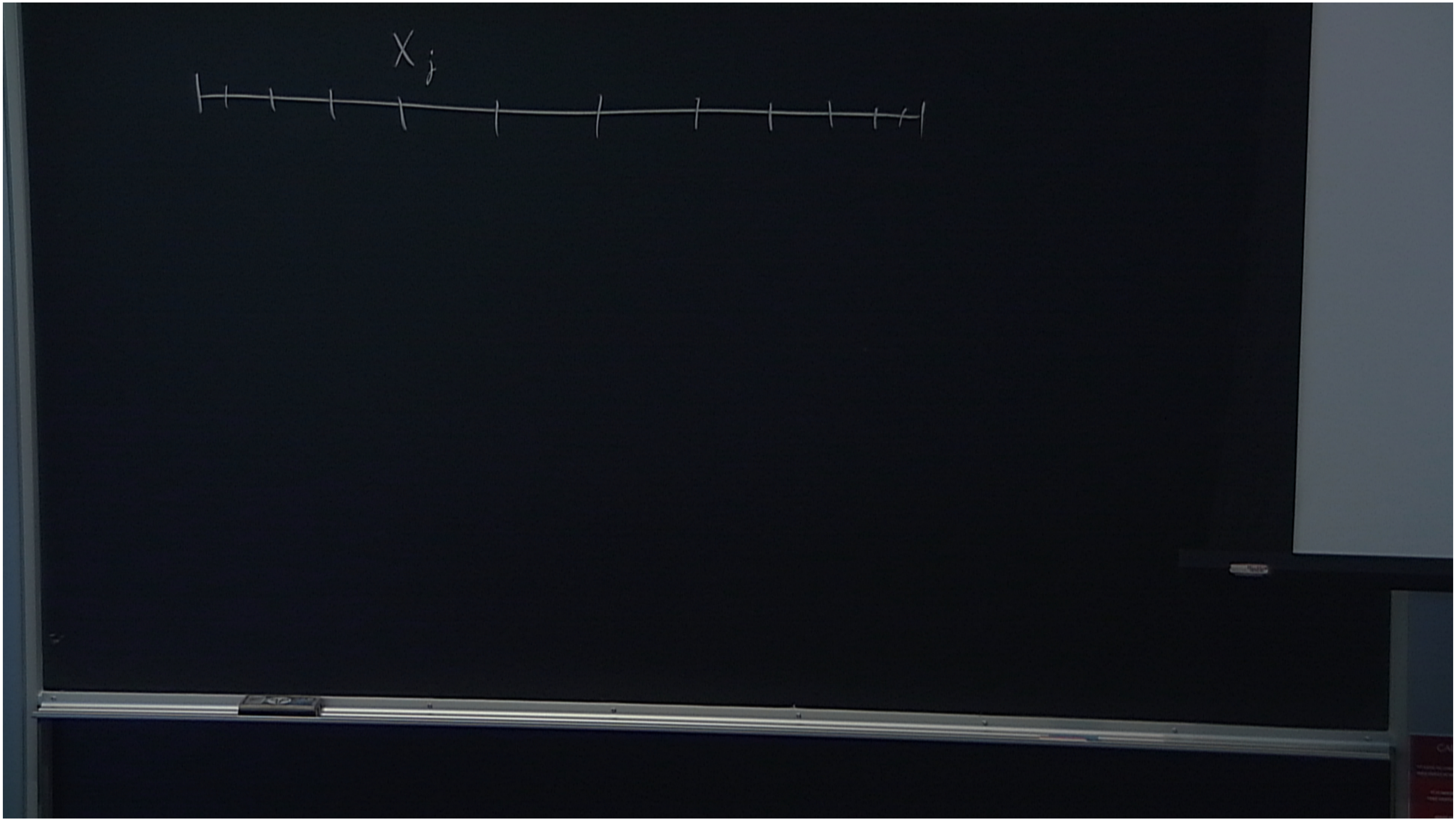
Summarising:

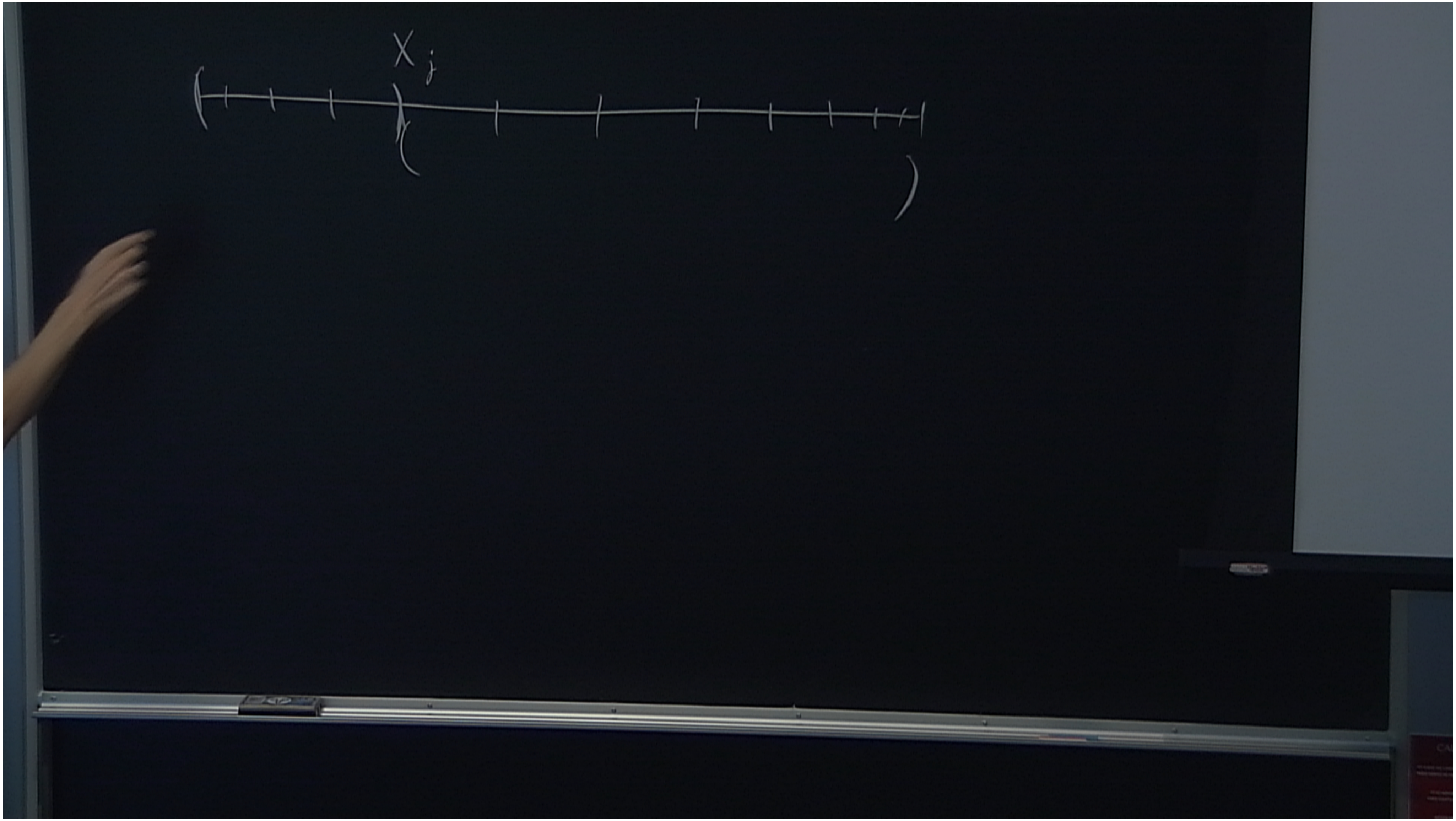
For Chebyshev pseudospectral method (or similar):

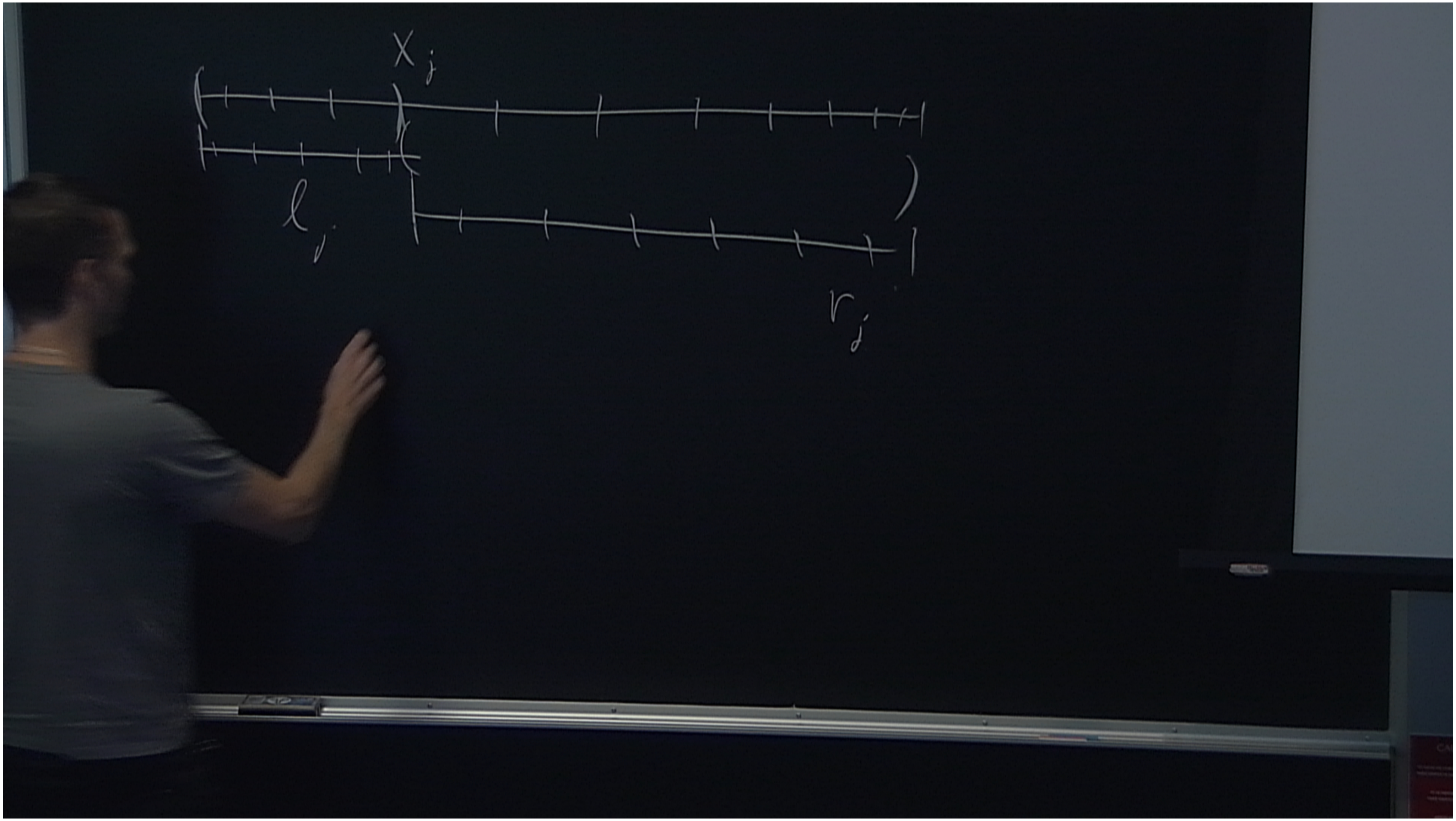
- must either find and decompose $\mathcal{L}_\mathrm{d}$ for $\mathrm{O}(n^2)$ or use spectral differentiation to find $\mathcal{L}_\mathrm{d} v$ for $\mathrm{O}(n \ln n)$;
- fundamental limitation is the conditioning of $\mathcal{L}_\mathrm{d}$.

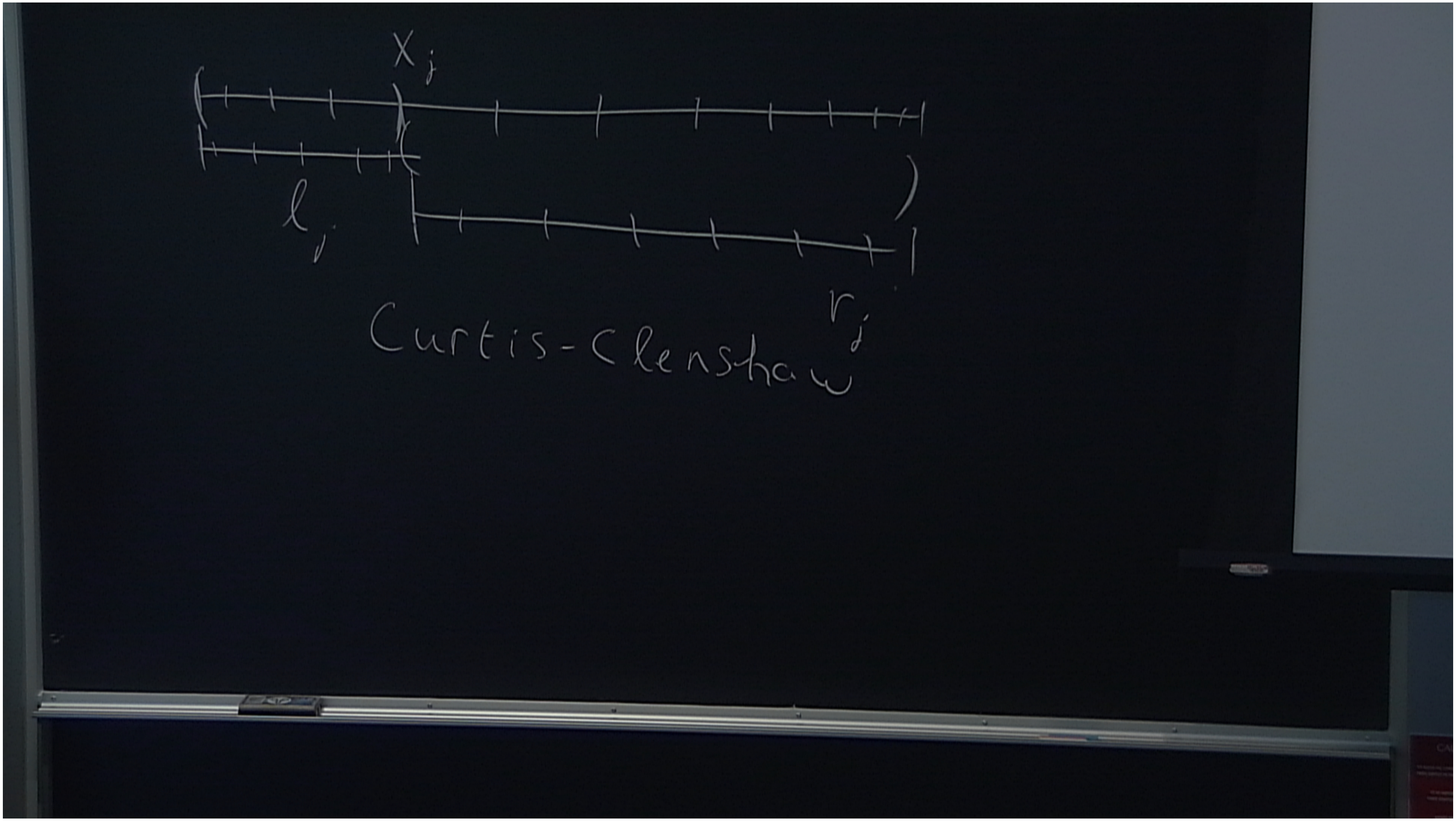
For Green's function based solving:

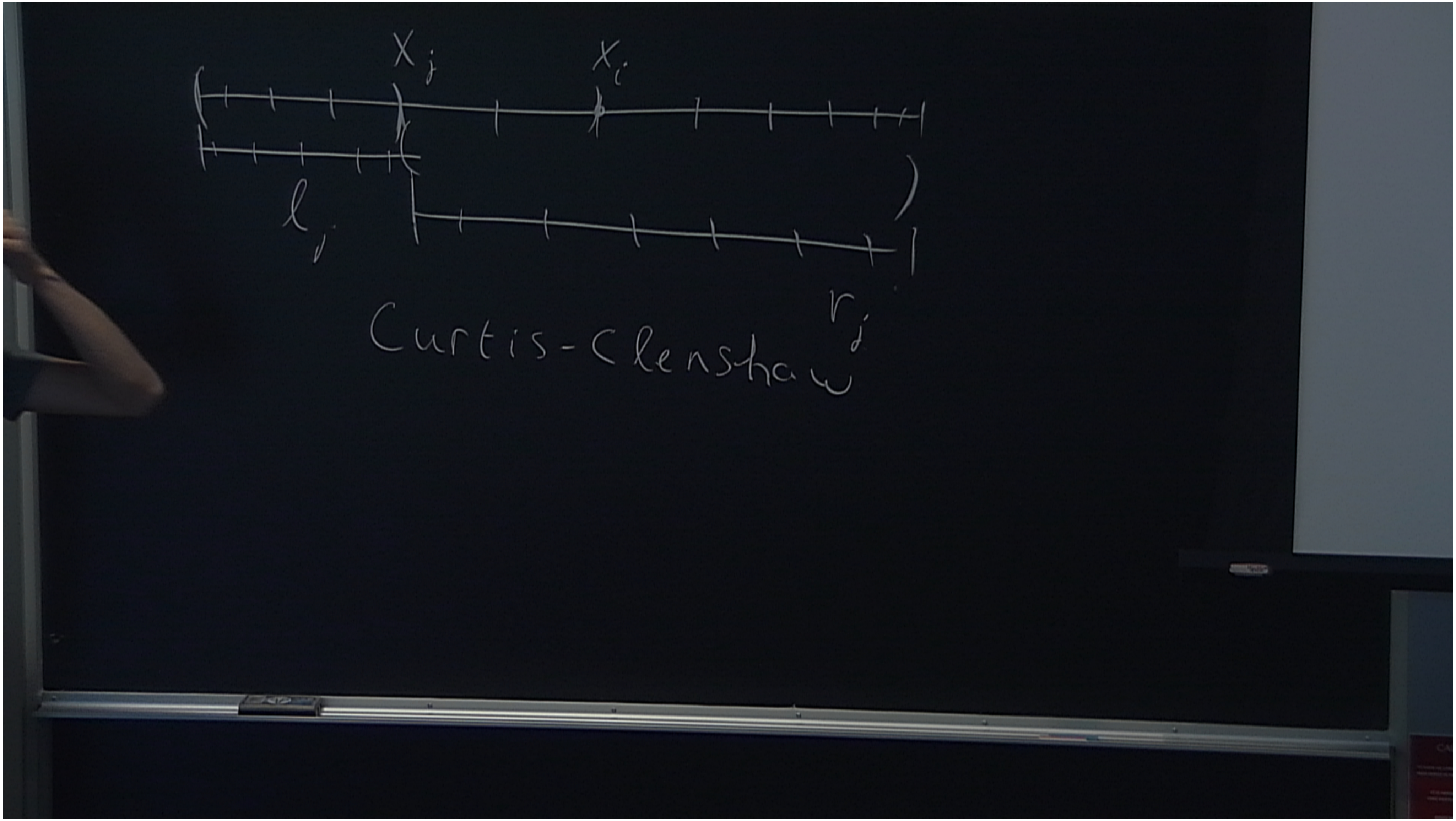
- must find a quadrature the is accurate even for the *finitely differentiable* integrand;
- fundamental limitation: ?

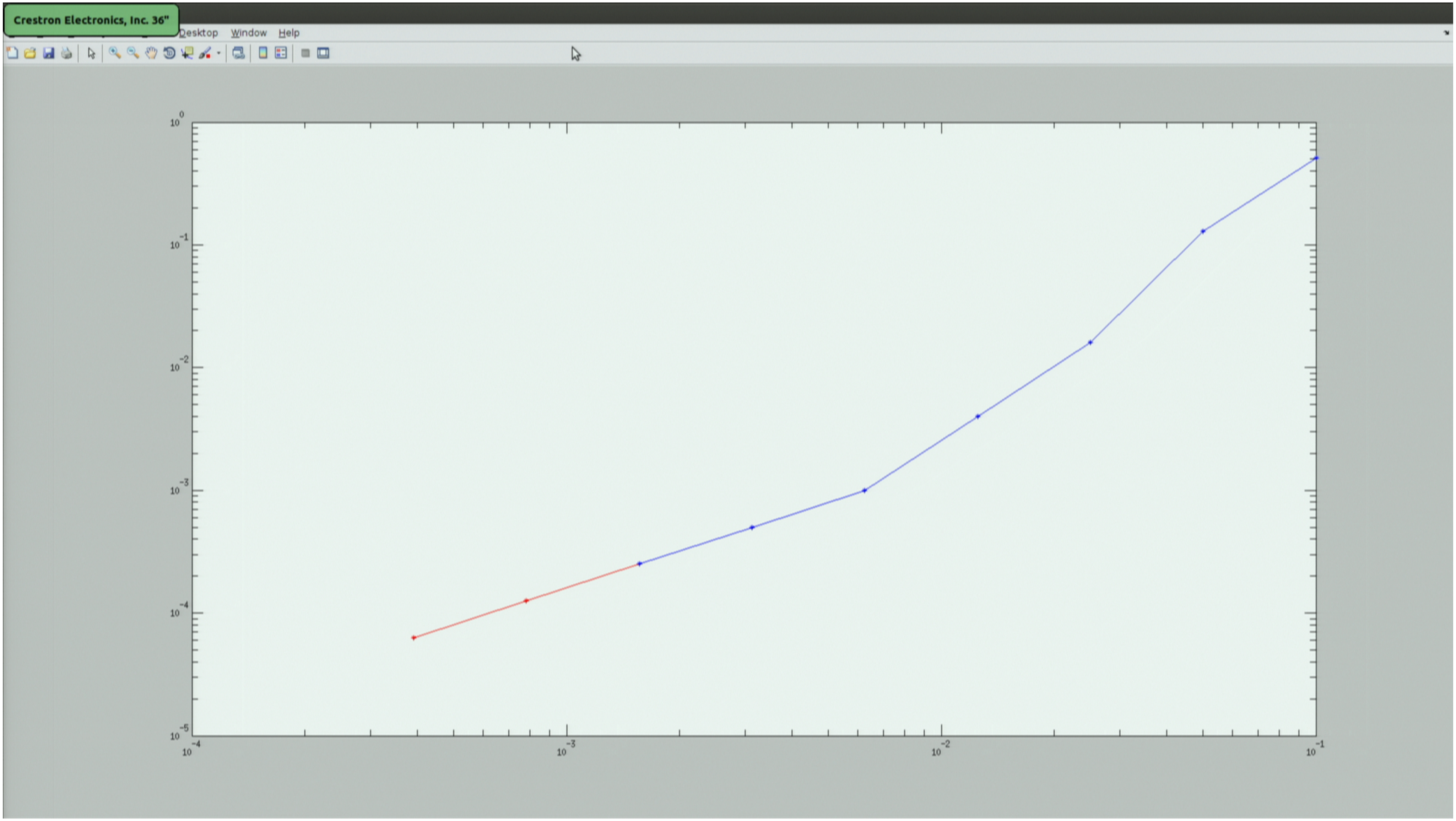

X_j


X j


x_j
ℓ_j
r_j


x_j
l_j
r_j
Curtis-Clenshaw


x_j
x_i
l_j
r_j
Curtis-Clenshaw


Crestron Electronics, Inc. 36"
Desktop
Window
Help
10^0
10^-1
10^-2
10^-3
10^-4
10^-5
10^-4
10^-3
10^-2
10^-1

Green's function – *invert first*

Both ideas are based on *discretizing first, then inverting*. With Green's function, we can *invert exactly, then discretize*:

$$u^{(k+1)} = \int_{y=-1}^{1} G(x,y)\left(u^k(y) + \Delta f(u^k(y))\right) \mathrm{d}y$$

where $G \in C^{p-2}$ satisfies

$$\mathcal{L}G(x,y) = \Delta(x-y)$$

The RHS now requires *numerical integration* instead of *differentiation*, which is much more stable.
The question is: what quadrature to use?

KS equation – Kuramoto-Sivashinsky IBVP

$$
\begin{aligned}
u_t + uu_x + u_{xx} + \nu\, u_{xxxx} &= 0 \\
-u(-1,t) = u(1,t) &= R \\
u_{xx}(-1,t) = u_{xx}(1,t) &= 0 \\
u(x,0) &= u_0(x)
\end{aligned}
$$

Set $\phi = Rx$, $u = v + \phi$, then

$$
\begin{aligned}
v_t &= \left[-\partial_x^2 - \nu\partial_x^4 - R\right] v + vv_x + \phi v_x + R\phi \\
v(-1,t) = v_{xx}(-1,t) &= v(1,t) = v_{xx}(1,t) = 0
\end{aligned}
$$

Crestron Electronics, Inc. 36"

KS Green's function – computing it

The linear operator

$$\mathcal{L} = \mathbb{I} + \Delta\partial_x^2 + \Delta\nu\partial_x^4 + \Delta R$$

is *symmetric* with these BC.
Spectrum:

Odd: $w_k^{\mathrm{o}} = \sin(k\pi x)$

$$\lambda_k^{\mathrm{o}} = 1 + \Delta R - \Delta\pi^2 k^2 + \Delta\nu\pi^4 k^4$$

Even: $w_k^{\mathrm{e}} = \cos\left(\left[k - \frac{1}{2}\right]\pi x\right)$

$$\lambda_k^{\mathrm{e}} = 1 + \Delta R - \Delta\pi^2\left(k - \frac{1}{2}\right)^2 + \Delta\pi^4\nu\left(k - \frac{1}{2}\right)^4$$

Eigenfunction expansion of G:

$$G(x,y) = \sum_{k=1}^{\infty} \left\{ \frac{w_k^{\mathrm{o}}(x)\, w_k^{\mathrm{o}}(y)}{\lambda_k^{\mathrm{o}}} + \frac{w_k^{\mathrm{e}}(x)\, w_k^{\mathrm{e}}(y)}{\lambda_k^{\mathrm{e}}} \right\}$$

Odd terms:

$$\sum_{k=1}^{\infty} \frac{w_k^{\mathrm{o}}(x)\, w_k^{\mathrm{o}}(y)}{\lambda_k^{\mathrm{o}}} = \frac{1}{2\Delta\nu(z_+ - z_-)} \sum_{k=1}^{\infty} \left\{ \frac{\cos(k\pi\,[x-y])}{\pi^2 k^2 - z_+} - \frac{\cos(k\pi\,[x-y])}{\pi^2 k^2 - z_-} - \frac{\cos(k\pi\,[x+y])}{\pi^2 k^2 - z_+} + \frac{\cos(k\pi\,[x+y])}{\pi^2 k^2 - z_-} \right\}$$

where

$$z_{\pm} = \frac{1}{2\nu} \pm \frac{1}{2\Delta\nu}\sqrt{\Delta^2 - 4\Delta\nu(1+\Delta R)} \text{ solve } \lambda_k^{\mathrm{o}} = 0 \text{ with } z = \pi^2 k^2$$

Eigenfunction expansion of G:

$$G(x,y) = \sum_{k=1}^{\infty} \left\{ \frac{w_k^{\mathrm{o}}(x) w_k^{\mathrm{o}}(y)}{\lambda_k^{\mathrm{o}}} + \frac{w_k^{\mathrm{e}}(x) w_k^{\mathrm{e}}(y)}{\lambda_k^{\mathrm{e}}} \right\}$$

Odd terms:

$$\sum_{k=1}^{\infty} \frac{w_k^{\mathrm{o}}(x) w_k^{\mathrm{o}}(y)}{\lambda_k^{\mathrm{o}}} = \frac{1}{2\Delta\nu(z_+ - z_-)} \sum_{k=1}^{\infty} \left\{ \frac{\cos(k\pi\,[x-y])}{\pi^2 k^2 - z_+} - \frac{\cos(k\pi\,[x-y])}{\pi^2 k^2 - z_-} - \frac{\cos(k\pi\,[x+y])}{\pi^2 k^2 - z_+} + \frac{\cos(k\pi\,[x+y])}{\pi^2 k^2 - z_-} \right\}$$

where

$$z_{\pm} = \frac{1}{2\nu} \pm \frac{1}{2\Delta\nu}\sqrt{\Delta^2 - 4\Delta\nu(1 + \Delta R)} \text{ solve } \lambda_k^{\mathrm{o}} = 0 \text{ with } z = \pi^2 k^2$$

Eigenfunction expansion of G:

$$G(x,y) = \sum_{k=1}^{\infty} \left\{ \frac{w_k^{\mathrm{o}}(x) w_k^{\mathrm{o}}(y)}{\lambda_k^{\mathrm{o}}} + \frac{w_k^{\mathrm{e}}(x) w_k^{\mathrm{e}}(y)}{\lambda_k^{\mathrm{e}}} \right\}$$

Odd terms:

$$\sum_{k=1}^{\infty} \frac{w_k^{\mathrm{o}}(x) w_k^{\mathrm{o}}(y)}{\lambda_k^{\mathrm{o}}} = \frac{1}{2\Delta\nu(z_+ - z_-)} \sum_{k=1}^{\infty} \left\{ \frac{\cos(k\pi\,[x-y])}{\pi^2 k^2 - z_+} - \frac{\cos(k\pi\,[x-y])}{\pi^2 k^2 - z_-} - \frac{\cos(k\pi\,[x+y])}{\pi^2 k^2 - z_+} + \frac{\cos(k\pi\,[x+y])}{\pi^2 k^2 - z_-} \right\}$$

where

$$z_{\pm} = \frac{1}{2\nu} \pm \frac{1}{2\Delta\nu}\sqrt{\Delta^2 - 4\Delta\nu(1+\Delta R)} \text{ solve } \lambda_k^{\mathrm{o}} = 0 \text{ with } z = \pi^2 k^2$$

…and after some mucking around:

$$\sum_{k=1}^{\infty} \frac{w_k^{o}(x) w_k^{o}(y)}{\lambda_k^{o}} =$$

$$\Re\left(\frac{1}{4\Delta\nu(z_+ - z_-)}\left\{\check{f}_+(x-y) - \check{f}_-(x-y) - \check{f}_+(x+y) + \check{f}_-(x+y)\right\}\right)$$

using the ISFT and

$$f_\pm = \frac{1}{\pi^2 k^2 - z_\pm}; \quad \check{f}_\pm(x) = -\frac{\cos(\pi p_\pm - \pi p_\pm |x|)}{\pi p_\pm \sin(\pi p_\pm)}, \quad \pi^2 p_\pm^2 = z_\pm$$

And similar for the even terms.

Finally:

$$G(x,y) = \frac{i}{\pi\Delta\nu(z_+ - z_-)} \Im\left(\frac{-\cos(2\pi p_+ - \pi p_+|x-y|) + \cos(\pi p_+|x+y|)}{p_+\sin(2\pi p_+)}\right)$$

Remarks:

- Assume that $\Delta < 4\nu/(1-4\nu R)$ so that $z_+ = z_-^*$.
- G is rewritten before numerical evaluation.
- G is smooth away from $x = y$.
- Use integration by parts for $\int G(x,y)u(y)u_y(y)\,\mathrm{d}y$

Simple idea:

- Make one master grid $\{x_j\}$.
- For each x_j, make two subgrids to cover $[-1, x_j]$ and $[x_j, 1]$, use quadrature separately on each.

The difficulty is that the sub grids are not *nested*.