FFT Solvers

Steffen Haug

The basic premise

Differentiation in time is *scaling* in frequency!

The basic premise

	t-**domain**	ω-**domain**
Definition	$f(t)$	$\hat{f}(\omega) = \int_{\mathbb{R}} f e^{-i\omega t}$
Differentiation	$\dfrac{df}{dt}$	$i\omega \hat{f}$

The basic premise

	t-domain	ω-domain
Definition	$f(t)$	$\hat{f}(\omega) = \int_{\mathbb{R}} f e^{-i\omega t}$
Differentiation	$\frac{df}{dt}$	$i\omega \hat{f}$
Second derivative	$\frac{d^2 f}{dt^2}$	$-\omega^2 \hat{f}$

The basic premise

	t-domain	ω-domain
Definition	$f(t)$	$\hat{f}(\omega) = \int_{\mathbb{R}} f e^{-i\omega t}$
Differentiation	$\frac{df}{dt}$	$i\omega \hat{f}$
Second derivative	$\frac{d^2 f}{dt^2}$	$-\omega^2 \hat{f}$
nth derivative	$\frac{d^n f}{dt^n}$	$(i\omega)^n \hat{f}$

You probably already memorized this as an undergraduate!

The simple idea

1. Estimate $\hat{f}$ with a FFT

$$\texttt{fft}\,(f)$$

The simple idea

1. Estimate $\hat{f}$ with a FFT
2. Differentiate by scaling with $(i\omega)^n$

$$(i\omega)^n \, \mathtt{fft}\,(f)$$

The simple idea

1. Estimate $\hat{f}$ with a FFT
2. Differentiate by scaling with $(i\omega)^n$
3. Compute the inverse FFT to obtain $\partial^n f / \partial t^n$

$$\frac{d^n f}{dt^n} \approx \texttt{ifft}\left((i\omega)^n \, \texttt{fft}\left(f\right)\right)$$

Why is this even interesting?

Assume an $N = n \times n$ grid.

Assume an $N = n \times n$ grid.

Most exact stencil methods solving an $N \times N$ linear system.

$O(N^3)$

Assume an $N = n \times n$ grid.

On the other hand, an $n \times n$ FFT can be calculated in $O(N \log N)$ operations.

Solvers are generally tailor made to a problem.

Exactly *how* the FFT is used may vary!

Solvers are generally tailor made to a problem.

I will show some examples of PDEs and how the FFT can be used.

“Direct integration”

$$\nabla^2 u = f$$

Poisson’s equation – boundary value problem

$$\mathscr{F}\left\{\nabla^2 u\right\} = \mathscr{F}\left\{f\right\}$$

$$\mathscr{F}\left\{\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right\}=\hat{f}$$

(Write out the laplacian)

$$\mathscr{F}\left\{\frac{\partial^2 u}{\partial x^2}\right\} + \mathscr{F}\left\{\frac{\partial^2 u}{\partial y^2}\right\} = \hat{f}$$

(Linearity)

$$-\omega^2 \hat{u} - \nu^2 \hat{u} = \hat{f}$$

(Derivative in frequency domain)

$$\hat{u}(\omega, \nu) = -\frac{1}{\omega^2 + \nu^2}\hat{f}$$

(Solve for $\hat{u}$)

ALGORITHM

$$F \quad \leftarrow \texttt{fft2}(f)$$

Compute a 2D FFT of the RHS.

Algorithm

$$F \leftarrow \texttt{fft2}(f)$$

$$U_{ij} \leftarrow -\frac{1}{{\delta^2}_{ij}} F_{ij}$$

"Integrate" by dividing by the frequencies.
(δ^2 obtained using `fftshift` on our frequencies)

ALGORITHM

$$
\begin{aligned}
F &\leftarrow \texttt{fft2}(f) \\
U_{ij} &\leftarrow -\frac{1}{{\delta^2}_{ij}} F_{ij} \\
u &\leftarrow \texttt{ifft2}(U)
\end{aligned}
$$

Recover the spatial signal.

When using a full DFT, we implicitly have periodic boundary conditions.

When using a full DFT, we implicitly have periodic boundary conditions.

If we are simulating a localized function, we can just pick a “big enough” box.

When using a full DFT, we implicitly have periodic boundary conditions.

Some problems are genuinely periodic!

When using a full DFT, we implicitly have periodic boundary conditions.

But *sometimes*, periodic BCs are undesirable.

When using a full DFT, we implicitly have periodic boundary conditions.

But *sometimes*, periodic BCs are undesirable.

For problems with only even order derivatives, we can do a “trick” to impose certain boundary conditions.

Assume some signal defined on $[0, L]$

Take the odd extension to $[-L, L]$ – all $\cos x$ terms vanish from the Fourier series!

Our solution will consist of sines, which are zero on the boundary of $[0, L]$!

For any period!

For any period!

This gives us the homogenous Dirichlet boundary condition $u = 0$ on the boundary.

Assume some signal defined on $[0, L]$

Take the *even* extension to $[-L, L]$ – all $\sin x$ terms vanish from the Fourier series!

Our solution will consist of cosines, with zero derivative on the boundary of $[0, L]$!

For any period!

For any period!

This gives us the homogenous von Neumann boundary condition $du/dx = 0$ on the boundary.

Simulation in frequency

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2}$$

Heat & diffusion equation - initial value problem

$$\frac{\partial \hat{u}}{\partial t} = -\omega^2 \hat{u}(\omega, t)$$

(FT in the spatial dimension x)

$$\frac{\partial \hat{u}}{\partial t} = -\omega^2 \hat{u}$$

(PDE in time and space converted into ODE in time and frequency)

$$\frac{\partial \hat{u}}{\partial t} = -\omega^2 \hat{u}$$

(Analytic solution well-known in this case, but that is generally not the case)

ALGORITHM

$$U_0 \leftarrow \texttt{fft}(u_0)$$

(Compute initial frequency distribution)

Algorithm

$$U_0 \leftarrow \texttt{fft}(u_0)$$

$$U \leftarrow \textbf{Integrate } U_0 \text{ with } \texttt{ode45}$$

(You can use other suitable ODE integrators)

Algorithm

$$
\begin{aligned}
U_0 &\leftarrow \texttt{fft}(u_0) \\
U &\leftarrow \textbf{Integrate } U_0 \text{ with } \texttt{ode45} \\
u &\leftarrow \texttt{ifft}(U)
\end{aligned}
$$

(Recover the desired spatial signal)

Expensive finite differences is replaced with relatively cheap FFT.

Expensive finite differences is replaced with relatively cheap FFT.

So... It's a free lunch?

Obviously these problems are cherry picked from those that are “nice” in the frequency domain.

Nonlinear PDEs

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \frac{\partial^2 u}{\partial x^2}$$

Nonlinear PDEs

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \frac{\partial^2 u}{\partial x^2}$$

Nonlinear PDEs

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \frac{\partial^2 u}{\partial x^2}$$

Products in time/space are convolutions in frequency.
$O(N^2)$

FFTs can still be an economical way to compute derivatives!

t-domain		**ω-domain**
Integration step	$\rightleftarrows$	Differentiate

But we have to move back and forth every integration step...

t**-domain** ω**-domain**

$$\text{Integration step} \rightleftarrows \text{Differentiate}$$

But we have to move back and forth every integration step...

FFTs are so fast that this might actually be sensible!

References

Anne Elster: *Parallelization issues and particle-in-cell codes*

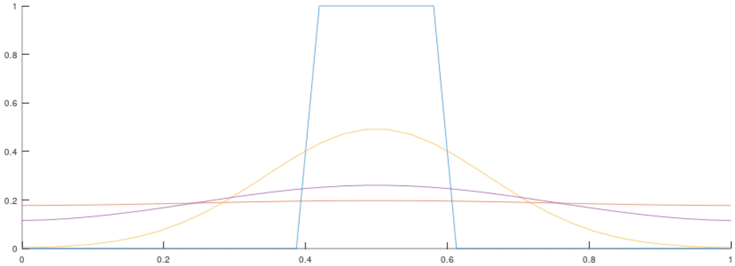
Chris Bretherton: *FFT-based 2D Poisson solvers*

Appendix

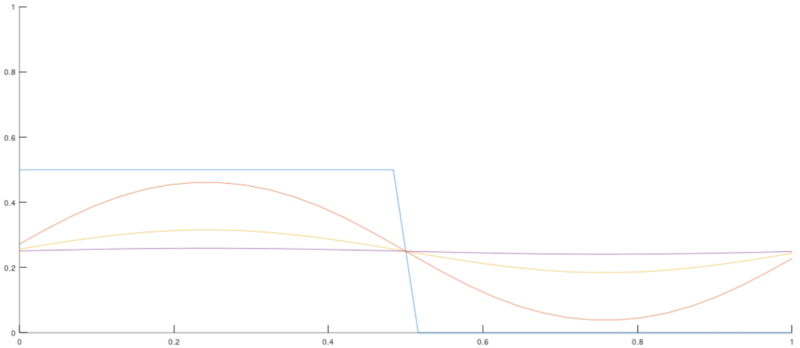
Example diffusion solver

(Matlab)

```
L = 1;  % Length of interval
N = 64; % Gridpoints
x = linspace(0, L, N);

% Fourier "frequencies":
k = (2*pi/L)*fftshift(-N/2 : N/2-1)';

%% Initial condition:
u0 = abs(x-L/2) <= 0.1;
U0 = fft(u0);

%% Solve with ODE-solver in the frequency domain
function dUdt = step(t, U, k)
    dUdt = -(k.^2) .* U;
end
[t, U] = ode45(@(t, U) step(t, U, k), 0:0.01:10, U0);

%% Recover the spatial signal from the ODE solution
for i = 1:length(t)
    u(i,:) = ifft(U(i,:));
end
```



A few snapshots of the diffusion. Note how because of the periodic BCs, all the "particles" is contained in the interval. A particle that goes out to the right, comes in from the left side, so the macroscopic distribution will flatten out to a certain constant level, and never reach zero as with the same equation solved over all of $\mathbb{R}$.



A few snapshots of the diffusion with the initial value `u0 = 0.5 * (x <= L/2)`. Here it is even more obvious that some particles are flowing out the left side and in the right side.