Developing high order, efficient, and stable time-evolution methods using a time-filtering approach

Sigal Gottlieb Chancellor Professor of Mathematics Center for Scientific Computing and Data Science Research University of Massachusetts Dartmouth

Holistic Design of Time-Dependent PDE Discretizations ICERM January 11, 2022

Joint work with: V. P. DeCaria (Bettis Atomic Power Laboratory), Z. J. Grant (ORNL) & W. J. Layton (UPitt)

Time filtering

- 2 Time Filters as General Linear Methods
- 3 Methods found using our GLM approach and optimization code
- 4 Numerical Results
- 5 Conclusions & Future Work

4 B K 4 B K

Time stepping methods with time-filtering

In many legacy codes, well-tested low-order time-stepping modules are deeply embedded and difficult to change.

Time-filters are a way to non-intrusively improve the time-stepping in legacy codes. Improvements include:

- Damp fluctuating scales without impacting smooth scales
- Reduce discrete curvature
- Enhance accuracy

Reformulating time-filtered methods as general linear methods, allows us to use optimization approaches and error inhibiting conditions to create highly accurate and (linearly) stable non-intrusive filters for legacy codes. Consider Layton's filtered implicit Euler scheme:

$$y^{(1)} = u^{n} + \Delta t F(y^{(1)})$$

$$u^{n+1} = y^{(1)} - \frac{1}{3} \left(u^{n-1} - 2u^{n} + y^{(1)} \right).$$

- First step is an implicit Euler step *first order, but has nice stability properties*
- Second step is a linear combination of some steps this is an anti-diffusive correction that also raises the order.

Adding the "filtering" step allows us to raise the order without touching the delicate and mysterious workings of the implicit legacy code.

We can generalize Layton's filtered Euler method by adding a "pre-processing" as well as a "post-processing" step:

$$\begin{aligned} \tilde{u}^n &= du^{n-1} + (1-d)u^n \\ y^{(1)} &= \tilde{u}^n + \Delta t F(y^{(1)}) \\ u^{n+1} &= \frac{1}{3-2d} \left(-u^{n-1} + 2(1-d)u^n + 2y^{(1)} \right). \end{aligned}$$

This extra freedom allows an entire family of second order methods that have implicit Euler as the driver.

• With d = 0, this becomes Layton's implicit Euler filter shown above:

$$y^{(1)} = u^{n} + \Delta t F(y^{(1)})$$

$$u^{n+1} = y^{(1)} - \frac{1}{3} \left(u^{n-1} - 2u^{n} + y^{(1)} \right).$$

• With d = 1, this becomes:

$$y^{(1)} = u^{n-1} + \Delta t F(y^{(1)})$$

$$u^{n+1} = 2y^{(1)} - u^{n-1}$$

Numerical Results

To compare the effect of these filters, consider the simple problem $\frac{dy}{dy} = -10y^2$:



Implicit Euler compared to Layton's filter with d = 0 and d = 1.

~ ··		-		
51	anl	(-01	-	ob.
J	ear	00		eD.

< □ > < □ > < □ > < □ > < □ > < □ >

General Linear Methods

How to I choose "good" values of the parameter d?

• Write the method as a GLM:

$$y^{(1)} = du^{n-1} + (1-d)u^n + \Delta t F(y^{(1)})$$

$$u^{n+1} = \frac{2d-1}{3-2d}u^{n-1} + \frac{4(1-d)}{3-2d}u^n + \frac{2}{3-2d}F(y^{(1)}),$$

- Construct an optimization code that optimizes certain stability properties, error constants, while treating the order conditions as constraints. Based on Ketcheson's MATLAB code that uses fmincon with sqp option.
- Optimizing this for linear order:

$$d=\frac{3-\sqrt{3}}{3}$$

gives a second order method that is third order for linear problem.

・ロト ・ 一下・ ・ 日下・

Methods with implicit Euler core

If we want to add more steps, we can get a third order accurate method based:

$$y^{(1)} = -\frac{1}{2}u^{n-2} + u^{n-1} + \frac{1}{2}u^{n}$$

$$y^{(2)} = y^{(1)} + \Delta tF\left(y^{(2)}\right)$$

$$u^{n+1} = \frac{5}{11}u^{n-2} - \frac{15}{11}u^{n-1} + \frac{15}{11}u^{n} + \frac{6}{11}y^{(2)}$$

- To identify the order of accuracy and stability, we write such a method as a GLM.
- This third order method is not A-stable: it $A(\alpha)$ region of stability with $\alpha \approx 71.51$.
- We gained an order of accuracy by allowing more steps, but we lost stability properties.

Producing Time Filters with a GLM approach

How do we we write such a method as a GLM? Consider the core method with s - 1 stages and k steps:¹

$$y_{1}^{n} = u^{n}$$

$$y_{i}^{n} = \sum_{\ell=1}^{k} \tilde{d}_{i\ell} u^{n-k+\ell} + \Delta t \sum_{\ell=1}^{k-1} \hat{a}_{i\ell} F(u^{n-k+\ell}) + \Delta t \sum_{j=1}^{s} a_{ij} F(y_{j}^{n}), \quad 2 \le i \le s$$

$$y_{i}^{n+1} = y_{s}^{n} = \sum_{\ell=1}^{k} \tilde{\theta}_{\ell} u^{n-k+\ell} + \Delta t \sum_{\ell=1}^{k-1} \tilde{b}_{\ell} F(u^{n-k+\ell}) + \Delta t \sum_{j=1}^{s} \tilde{b}_{j} F(y_{j}^{n})$$

Note that the final row coefficients are the same as the prior row coefficients

$$ilde{ heta}_{I}= ilde{d}_{s\ell}, \;\; ilde{b}_{\ell}=\hat{a}_{s\ell}, \;\; ilde{b}_{j}=a_{sj}.$$

¹This is John Butcher's notation

L

Pre-filters and post-filters using a GLM approach

• To pre-process the method, we modify u^n . To do this, we change the initial stage so that u^n is replaced by a linear combination of the old steps

$$y_1^n = u^n \longrightarrow y_1^n = \sum_{\ell=1}^k d_{1\ell} u^{n-k+\ell}$$

and then use y₁ⁿ instead of uⁿ in all the middle stages of the core method. This propagates throughout the GLM form of the method.
To post-process the method, we simply modify the final line to

$$u^{n+1} = \sum_{\ell=1}^{k} \theta_{\ell} u^{n-k+\ell} + \Delta t \sum_{\ell=1}^{k-1} \hat{b}_{\ell} F(u^{n-k+\ell}) + \Delta t \sum_{j=1}^{s} b_{j} F(y_{j}^{n})$$

where the coefficients θ_{ℓ} , \hat{b}_{ℓ} , b_j can be chosen freely. Choose these coefficients so the method is non-intrusive.

イロト 不得 トイラト イラト 一日

Time filtering using a GLM approach

The overall method can now be written as

$$y_{1}^{n} = \sum_{\ell=1}^{k} d_{1\ell} u^{n-k+\ell}$$
$$y_{i}^{n} = \sum_{\ell=1}^{k} d_{i\ell} u^{n-k+\ell} + \Delta t \sum_{\ell=1}^{k-1} \hat{a}_{il} F(u^{n-k+\ell}) + \Delta t \sum_{j=1}^{s} a_{ij} F(y_{j}^{n}) \quad 2 \le i \le s$$
$$u^{n+1} = \sum_{\ell=1}^{k} \theta_{\ell} u^{n-k+\ell} + \Delta t \sum_{l=1}^{k-1} \hat{b}_{\ell} F(u^{n-k+\ell}) + \Delta t \sum_{j=1}^{s} b_{j} F(y_{j}^{n}).$$

where $d_{i\ell} = \tilde{d}_{i1}d_{1\ell} + \tilde{d}_{i\ell}$ for $\ell = 1, k - 1$, and $d_{ik} = \tilde{d}_{i1}d_{1k}$, where the pre-processing coefficients $d_{1\ell}$ are chosen freely. and $d_{1\ell}$, θ_{ℓ} , \hat{b}_{ℓ} , b_{j} can be chosen for all ℓ and j.

,

What's the point?

- Approach the design of the pre- and post-processor as an optimization problem (we coded it in MATLAB):
 - Given the coefficients of the core method, we treat $d_{1\ell}$, θ_ℓ , \hat{b}_ℓ , b_j as free parameters
 - The equality constraints are given by the order conditions we wish the new overall GLM to satisfy
 - Additional requirements (low storage etc.) can be imposed on the structure of the coefficients
 - Define our objective function:
 - a maximized linear stability region
 - 2 positivity preserving properties
 - Small error constants
 - reducing discrete fluctuations or curvature
 - efficient implementation
- New GLMs based on implicit Euler, trapezoid rule, midpoint rule, BDF methods, fully implicit Runge–Kutta

・ ロ ト ・ 同 ト ・ 三 ト ・ 三 ト

We can filter BDF2 which is second order and A-stable

$$y^{(1)} = d_1 u^{n-3} + d_2 u^{n-2} + d_3 u^{n-1} + d_4 u^n \quad (\text{Pre-filter})$$

$$y^{(2)} = -\frac{1}{3} u^{n-1} + \frac{4}{3} y^{(1)} + \frac{2}{3} \Delta t F(y^{(2)}) \quad (\text{BDF2})$$

$$u^{n+1} = \theta_1 u^{n-3} + \theta_2 u^{n-2} + \theta_3 u^{n-1} + \theta_4 u^n + b \Delta t F(y^{(2)}) \quad (\text{Post-filter})$$

This is a third order (stage order q = 2) four step three stage method with linear stability region $A(\alpha)$ with $\alpha = 89.624719317072476$.

Filtering BDF2 to third order con't

Sigal

The method is $A(\alpha)$ stable with $\alpha = 89.624719317072476$. The linear stability region is:



Gottlieb	Time-stepping methods		ICERM

イロト イポト イヨト イヨト

15 / 32

Filtered Trapezoid Rule

Use the implicit trapezoid rule as the core method

$$y^{(1)} = u^{n}$$

$$y^{(2)} = u^{n} + \frac{1}{2}\Delta tF(y^{(1)}) + \frac{1}{2}\Delta tF(y^{(2)})$$

$$u^{n+1} = y^{(2)}$$

The filtered method takes the form

$$y^{(1)} = \sum_{\ell=1}^{k} d_{1\ell} u^{n-k+\ell}$$

$$y^{(2)} = y^{(1)} + \frac{1}{2} \Delta t F(y^{(1)}) + \frac{1}{2} \Delta t F(y^{(2)})$$

$$u^{n+1} = \sum_{\ell=1}^{k} \theta_{\ell} u^{n-k+j} + 2\hat{b}(y^{(2)} - y^{(1)}).$$

Sigal Gottlieb

Filtered Trapezoid Rule

In the optimization code we can require the filtered scheme (with three or four steps) to satisfy third or fourth order accuracy conditions, to produce the P^3 filter:

- $d_1 = -0.102545818304193$ θ
- $d_2 = 0.493175701880453$

$$d_3 = 0.609370116423740$$

$$\hat{b} = 0.721809600017651$$

$$\theta_1 = -0.027735512184323$$

 $\theta_2 = 0.499090224403948$

$$n_2 = 0.499090224403948$$

$$\theta_3 = -0.528645287780375$$

and the P^4 filter

 $d_1 = 0.105235702866242$ θ_1 θ_2

$$d_2 = -0.433307797383908$$

$$d_3 = 0.929258897474200$$

$$d_4 = 0.398813197043466$$

$$\hat{b} = 0.868632639137240$$

$$=$$
 0.0204387588888459

$$= -0.141286522358621$$

$$\theta_3 = 0.958522046326348$$

$$\theta_4 = 0.162325717143815$$

- 本間下 本臣下 本臣下 三臣

In the optimization code we can require the filtered scheme (with three or four steps) to satisfy third or fourth order accuracy conditions, to produce the P^3 or P^4 filters. On the same problem as before, we verify the order:

Δt	Trapezoid Rule		$+ P^3$ Filter		$+ P^4$ Filter	
5.00e-03	4.41e-06	-	5.02e-07	-	1.24e-07	-
2.50e-03	1.25e-06	1.81	7.47e-08	2.74	9.25e-09	3.74
1.25e-03	3.36e-07	1.89	1.02e-08	2.86	6.35e-10	3.86
8.33e-04	1.53e-07	1.93	3.14e-09	2.91	1.29e-10	3.91
6.25e-04	8.73e-08	1.95	1.34e-09	2.94	4.17e-11	3.94

The cost of this increase accuracy is loss of A-stability.

Filtered Trapezoid Rule



Sigal Gottlieb

ICERM 19 / 32

Filtered fully implicit Runge–Kutta (2,2)

We want a third order A-stable method, and can obtain it from filtering the fully implicit Runge–Kutta (2,2):

Consider the L-stable fully implicit Lobatto IIIC scheme:

$$y^{(1)} = u^{n} + \frac{1}{2}\Delta tF(y^{(1)}) - \frac{1}{2}\Delta tF(y^{(2)})$$
$$u^{n+1} = u^{n} + \frac{1}{2}\Delta tF(y^{(1)}) + \frac{1}{2}\Delta tF(y^{(2)})$$

The 2 Step time-filtered scheme can be written as :

$$\hat{u} = 0.373461706729200u^{n-1} + 0.626538293270800u^{n}$$

$$y^{(1)} = \hat{u} + \frac{1}{2}\Delta tF(y^{(1)}) - \frac{1}{2}\Delta tF(y^{(2)})$$

$$y^{(2)} = \hat{u} + \frac{1}{2}\Delta tF(y^{(1)}) + \frac{1}{2}\Delta tF(y^{(2)})$$

$$u^{n+1} = a_{1}u^{n-1} + a_{2}u^{n} + a_{3}y^{(1)} + a_{4}y^{(2)}$$

where $a_1 = -0.075425887737539$ $a_2 = 0.551112405533260$ $a_3 = -0.596071637983322$ $a_4 = 1.120385120187601$. This scheme is third order and A-stable but not L-stable.

Sigal Gottlieb

Time-stepping methods

ICERM 20 / 32

Third order EIS method based on backward Euler

Can we get a third order A-stable method based on implicit Euler? To do this, we designed an alternating-filter implicit Euler method, that can be implemented within a legacy code²

$$y_n^{(1)} = \frac{23}{5}y_{n-1}^{(2)} - 3u^n - \frac{9}{5}y_{n-1}^{(1)} + \frac{6}{5}y_{n-1}^{(3)}$$

$$y_n^{(2)} = y_n^{(1)} + \Delta tF(y_n^{(2)})$$

$$y_n^{(3)} = \frac{5}{12}u^n - \frac{1}{12}y_n^{(2)} - \frac{5}{12}y_{n-1}^{(3)} + \frac{13}{12}y_n^{(1)}$$

$$u^{n+1} = y_n^{(3)} + \Delta tF(u^{n+1}).$$

This method is A-stable and the implicit stages are simply backward Euler. Taylor expansions predict that this method is second order, but we designed it to be error inhibiting so it will be third order.

Error inhibiting schemes: how do they work?

GLMs can be written as:

$$V^{n+1} = DV^n + \Delta t AF(V^n) + \Delta t RF(V^{n+1}),$$

$$\tau^n = \sum_{j=1}^{\infty} \tau_j \Delta t^j \frac{d^j u(t_n)}{dt^j} \text{ so that where}$$

$$\tau_j = \frac{1}{j!} c^j - \frac{1}{j!} D(c - e)^j - \frac{1}{(j-1)!} A(c - e)^{j-1} - \frac{1}{(j-1)!} Rc^{j-1}$$

and c is the vector of abscissas $(c_1, c_2, ..., c_s)$ and e is the vector of ones. If $\tau_j = 0$ for j = 1, ..., p then

$$\tau^n = O(\Delta t^{p+1}) \Longrightarrow E_n = O(\Delta t^p).$$

Sigal Gottlieb

▲□▶ ▲圖▶ ▲ 臣▶ ▲ 臣▶ 三臣 - のへで

Error inhibiting schemes: how do they work?

The truncation error enters into the next time-step through the formula

$$V^{n+1} = DV^n + \Delta tAF(V^n) + \Delta tRF(V^{n+1}),$$

and usually the truncation error accumulates so that the $\tau^n = O(\Delta t^{p+1})$ grows over $\frac{1}{\Delta t}$ timesteps to become $O(\Delta t^p)$.

However, if τ_{p+1} lives in the nullspace of D, then at every step the truncation error term will be annihilated $D\tau_{p+1} = 0$, so the first term in the truncation error will never be allowed to accumulate, and the global error will be:

$$E_n = O(\Delta t^{p+1}).$$

(Quasi-consistency theory of Kulikov and Wiener, etc.; Error inhibiting methods by Ditkowski, Gottlieb, and Grant.)

Sigal Gottlieb

イロト 不得 トイラト イラト 一日

Third order EIS method based on backward Euler

$$y_n^{(1)} = \frac{23}{5}y_{n-1}^{(2)} - 3u^n - \frac{9}{5}y_{n-1}^{(1)} + \frac{6}{5}y_{n-1}^{(3)}$$

$$y_n^{(2)} = y_n^{(1)} + \Delta tF(y_n^{(2)})$$

$$y_n^{(3)} = \frac{5}{12}u^n - \frac{1}{12}y_n^{(2)} - \frac{5}{12}y_{n-1}^{(3)} + \frac{13}{12}y_n^{(1)}$$

$$u^{n+1} = y_n^{(3)} + \Delta tF(u^{n+1}),$$

can we written as

$$y_n^{(1)} = \frac{14}{5}u^{n-\frac{1}{3}} - \frac{9}{5}u^n + \frac{9}{5}\Delta tF(u^{n-\frac{1}{3}}) - \frac{6}{5}\Delta tF(u^n)$$
$$u^{n+\frac{2}{3}} = y_n^{(2)} = y_n^{(1)} + \Delta tF(y_n^{(2)})$$
$$y_n^{(3)} = y_n^{(2)} + \frac{5}{12}\Delta tF(u^n) - \frac{13}{12}\Delta tF(y_n^{(2)})$$
$$u^{n+1} = y_n^{(3)} + \Delta tF(u^{n+1}).$$

Sigal Gottlieb

ICERM 24 / 32

Third order EIS method based on backward Euler

This method can be written using a block method approach: $V^n = \left[u^{n-1/3}; u^n \right]$

$$V^{n+1} = \begin{pmatrix} \frac{14}{5} & -\frac{9}{5} \\ \frac{14}{5} & -\frac{9}{5} \end{pmatrix} V^n + \Delta t \begin{pmatrix} \frac{9}{5} & -\frac{6}{5} \\ \frac{9}{5} & -\frac{47}{60} \end{pmatrix} F(V^n) + \Delta t \begin{pmatrix} 1 & 0 \\ -\frac{12}{5} & 1 \end{pmatrix} F(V^{n+1})$$

Truncation error analysis (Taylor series expansions) shows that we can expect second order solutions, but the truncation error vector lives in the nullspace of the matrix

$$\left(\begin{array}{cc} \frac{14}{5} & -\frac{9}{5} \\ \frac{14}{5} & -\frac{9}{5} \end{array}\right)$$

so this method gives us third order accurate results.

This is an error inhibiting method.

Sigal	Gottlieb
-------	----------

How do these methods perform in practice?

- We tested these with finite element code for the Navier-Stokes equations.
- This set of problems with Reynolds numbers between 120 and 600.
- The EIS-3 method was most stable (apparently A-stability matters)
- The EIS-3 method was very accurate (similar to third order methods, when they converged)

Numerical Results: Simulations by Victor DeCaria (ORNL).

Numerical Results: Flow past a cylinder

Benchmark test with kinematic viscosity $\nu = 10^{-3}$, final time $T_f = 8$, and domain

$$\Omega = \left\{ (x,y) | 0 < x < 2.2, 0 < y < 0.41, \text{ and } (x - 0.2)^2 + (y - 0.2)^2 > 0.05^2 \right\}.$$

Δt	$t(c_{d,\max})$	c _{d,max}	$t(c_{l,max})$	c _{l,max}	$\Delta p(T_f)$	
Reference Values						
—	3.93625	2.950921575	5.693125	0.47795	-0.1116	
			IE			
0.005	3.93	2.950301672	6.28	0.17604	-0.1005	
0.0025	3.9325	2.950371110	6.215	0.30336	-0.1070	
0.00125	3.93375	2.950529384	5.7175	0.38229	-0.1114	
		IE-Pre-2 (one IE solve a	and one filter per tir	nestep)		
0.005	3.935	2.950802171	5.72	0.45978	-0.1111	
0.0025	3.935	2.950872330	5.7	0.47413	-0.1120	
0.00125	3.93625	2.950889791	5.695	0.47728	-0.1117	
IE-Filt($\frac{3-\sqrt{3}}{2}$) (one IE solve and two filters per timestep)						
0.005	3.93	2.950839424	5.71	0.46722	-0.1127	
0.0025	3.9325	2.950880844	5.695	0.47567	-0.1124	
0.00125	3.935	2.950891744	5.6925	0.47762	-0.1120	
IE-Pre-Post-3 (one IE solve and two filter per timestep)						
0.005	7.825	435.1275230	7.84	205.33324	-5.1966	
0.0025	3.935	2.950897874	5.6925	0.47895	-0.1116	
0.00125	3.93625	2.950895596	5.6925	0.47833	-0.1116	
IE-EIS-3 (two IE solves and two filters per timestep)						
0.01	3.93667	2.950816639	5.69667	0.46183	-0.1119	
0.005	3.935	2.950884378	5.69333	0.47608	-0.1117	
0.0025	3.93667	2.950893844	5.6925	0.47797	-0.1116	

(*IE-pre-post-3 did not converged for several time-step sizes.)

◆□▶ ◆圖▶ ◆厘▶ ◆厘≯

Sigal Gottlieb

ICERM 27

27 / 32

Numerical Results: Offset Cylinder Test

A body forced internal flow between two offset cylinders: the domain is a unit cylinder, and there is a small cylinder inside.

(*IE-pre-post-3 did not converge for several time-step sizes.)



Figure 7: For $\Delta t = 0.0025$, IE-Pre-Post-3 is unstable while IE-EIS-3 performs the best. For $\Delta t = 0.00125$, IE-Pre-Post-3 becomes stable and is the most accurate solution.

Numerical Results: Offset Cylinder Test



Conclusions

- Time filtering is an approach that can be helpful to improve the accuracy of methods within legacy codes (and maybe for other reasons as well)
- We re-wrote the time-filtering procedure as a GLM
- This approach can be applied to multistep, multi-stage (Runge–Kutta), or multistep multi-stage methods
- We constructed an optimization code that can find the coefficients of filters with a variety of desirable properties
- Combining this approach with the error inhibiting approach led to a third order A-stable method
- We found methods that perform well on some challenging fluids problems tested by Layton and DeCaria (to appear in JCP).

(4) (日本)

- Use this GLM optimization approach to find methods with other desirable properties (e.g. positivity, SSP, energy conservation)
- Consider nonlinear filtering: can we write this as a GLM? (not exactly)
- Time-filtering with variable stepsizes
- EIS time-filters with post-processing (Following work by Ditkowski, Gottlieb, and Grant)
- Turning the time-filters on and off
- Designing more alternating filters (maybe for BDF2)

Thank you!

Sigal Gottlieb

Time-stepping methods

■ ► ■ つへの ICERM 32/32

イロト イヨト イヨト イヨト