Notes on the BENCHOP implementations for the FDAD method

Lina von Sydow (lina@it.uu.se)

March 3, 2015

Abstract

This text describes the FD-AD method and its implementation for the BENCHOP-project.

1 Spatial discretization

The problems considered are all on the form

$$\frac{\partial u}{\partial t} + \mathcal{L}u = 0 \quad , \quad t \in [0, T] \tag{1}$$

where $\mathcal{L}$ is a partial-(integro) operator in one or two spatial dimensions.We will describe the spatial discretization with adaptivity for a one-dimensional problem in s, . The generalization to a two-dimensional problem is straight-forward and can be found in , and .

We discretize (1) on an equidistant grid s_j using centered, second-order finite differences such that for a computed solution $u_h \in C^2$ it holds

$$u_h = u + h^2 c(s) \tag{2}$$

after neglecting high-order terms and hence $u_{2h} = u + (2h)^2 c(s)$. Using the second-order accuracy also in the local discretization error in space τ_h we get

$$\tau_h = h^2 \eta(s). \tag{3}$$

From the definition of the local truncation error $\tau_h = \mathcal{L}_h u - \mathcal{L}u$ and (2) we get

$$\tau_h = \mathcal{L}_h u_h - \mathcal{L}u - h^2 \mathcal{L}_h c(x) \quad , \quad \tau_{2h} = \mathcal{L}_{2h} u_h - \mathcal{L}u - h^2 \mathcal{L}_{2h} c(x), \tag{4}$$

where the term $\mathcal{L}_{2h} u_h$ is defined as the operator $\mathcal{L}_{2h}$ acting on every second element in u_h. Subtracting the first equation in (4) from the second, and defining $\delta_h = \mathcal{L}_h u_h$ and $\delta_{2h} = \mathcal{L}_{2h} u_h$ gives

$$\tau_{2h} - \tau_h = \delta_{2h} - \delta_h - h^2 (\mathcal{L}_{2h} - \mathcal{L}_h) c(x) = \delta_{2h} - \delta_h + \mathcal{O}(h^4).$$

Now using (3) and omitting high-order terms we get

$$\eta(s) \approx \frac{\delta_{2h} - \delta_h}{3h^2} \quad , \quad \tau(s) = \frac{\delta_{2h} - \delta_h}{3}, \tag{5}$$

i.e. we can estimate $\eta(s)$ by computing a solution $u_{\bar{h}}$ using the spatial discretization $\bar{h}$ and employ (5). If we require $|\tau_h| = |h^2\eta(x)| < \epsilon$ for some tolerance ϵ we can obtain this by computing a solution using the new spatial discretization $h(x)$ defined by

$$h(s) = \bar{h}\sqrt{\frac{\epsilon}{|\tau_{\bar{h}}(s)|}}.$$

To prevent us from using too large spatial steps, we introduce a small parameter d and define

$$h(x) = \bar{h}\sqrt{\frac{\epsilon}{|\tau_{\bar{h}}(s)| + \epsilon \cdot d}}. \tag{6}$$

We use extrapolation of $\tau_{\bar{h}}$ close to the boundaries $s = s_{\min}$, $s = s_{\max}$ and $v = v_{\max}$ to remove the effects caused by the boundary conditions used. To ensure a smooth $\tau_{\bar{h}}$ we perform q smoothing iterations according to

$$\tau_{\bar{h}}(s_k) = \left(\tau_{\bar{h}}(s_{k-1}) + 2\tau_{\bar{h}}(s_k) + \tau_{\bar{h}}(s_{k+1})\right)/4.$$

Since (1) is time-dependent the local discretization error τ_h will vary in time. We will use the solution u_h at three different time-steps 0, $T/3$, and $2T/3$ and use $\max|\tau_h|$ over these time-steps when we compute the new computational grids.

We end this section by summarizing the algorithm for adaptivity as follows:

1. Compute a solution using a coarse spatial grid with N_c grid-points in space and a coarse temporal discretization with M_c time-steps.
2. Estimate the local truncation error on this grid and compute a new spatial grid using (6) for some given ϵ.
3. Compute a new solution using the new spatial grid with N_f grid-points in space and M_f time-steps.

2 Temporal discretization

The spatial discretization described in Section 1 leads to the system of ordinary differental equations

$$\frac{du_h}{dt} + A_h u_h = 0, \tag{7}$$

where A_h for a one-dimensional problem is a tri-diagonal matrix of size $N \times N$. For most benchmarking problems we have used discontinuous Galerkin in time to solve (7), and when it for some reason didn't compute accurate solutions, we used BDF-2.

2.1 Discontinuous Galerkin

The time-interval $[0,T]$ is partitioned into M subintervals $\{I_m = (t_{m-1}, t_m)\}_{m=1}^{M}$ of size $k = t_m - t_{m-1} = \frac{T}{M}$. Define $\mathcal{P}^r(I_m)$ as the space of polynomials of degree r or less on the interval I_m and $\mathbb{U} = \{U : U_m \in \mathcal{P}^r(I_m)\}$ to be the finite element space containing the piecewise polynomials. The solution U is continuous within each time interval I_m, but may be discontinuous at the nodes $t_1, \ldots, t_{M-1}$. We define the one-sided limits of a piecewise continuous function $u(t)$ as $u_m^+ := \lim_{v\to 0^+} u(t_m + v)$, $u_m^- := \lim_{v\to 0^+} u(t_m - v)$, and the "jump" in $u(t)$ across t_m as $[u_m] := u_m^+ - u_m^-$.

The dG method of degree r (dG(r) to solve (7) reads as follows: Find $U \in \mathbb{U}$, satisfying $U_0^- = u_0$, such that $\sum_{m=1}^{M} \int_{I_m} (\dot{U}_m - AU_m)w(t)\,dt + \sum_{m=1}^{M} [U_{m-1}]w(t_{m-1}) = 0$ for all $w(t) \in \mathbb{U}$. In practice U can be computed in each interval

$$\int_{I_m} (\dot{U}_m - AU_m)w(t)\,dt + [U_{m-1}]w(t_{m-1}) = 0 \tag{8}$$

for $m = 1, \ldots, M$. Let $\{\varphi\}_{j=0}^{r_m}$ be a basis of the polynomial space $\mathcal{P}_{r_m}(-1,1)$ and let time shape functions on time interval I_m be given by $\varphi_j \circ F_m^{-1}$, where the mapping $F_m : (-1,1) \to I_m$ is given by $t = F_m(x) = \frac{1}{2}(t_{m-1} + t_m) + \frac{1}{2}kx, \quad x \in (-1,1)$. Since the dG approximation U_m in each time interval I_m is in the polynomial space $\mathcal{P}_{r_m}(I_m)$, it can uniquely be expressed in the basis $\{\varphi\}_{j=0}^{r_m}$ as $U_m = \sum_{j=0}^{r_m} u_{m,j}(\varphi_j \circ F_m^{-1})$. Inserting this into (8), and letting the test function $w(t)$ be the basis $\{\varphi\}_{j=0}^{r_m}$, we get after some algebraic manipulations

$$\sum_{i,j=0}^{r_m} \left(C_{ij} - \frac{k}{2} G_{ij} \cdot A \right) u_{m,j} = \sum_{i=0}^{r_m} f_{m,i}, \tag{9}$$

with $f_{m,i} = \varphi_i(-1) \sum_{j=0}^{r_m} \varphi_j(1) u_{m-1,j}$, $C_{ij} = \int_{-1}^{1} \varphi_j' \varphi_i \, d\tau + \varphi_j(-1)\varphi_i(-1)$, $G_{ij} = \int_{-1}^{1} \varphi_j \varphi_i \, d\tau$. Dropping the subscript m for sake of readability and representing (9) in matrix form results in

$$\left(\mathbf{C} \otimes \mathbf{I} - \frac{k}{2} \mathbf{G} \otimes \mathbf{A} \right) \mathbf{u} = \mathbf{f}, \tag{10}$$

where $\otimes$ is the Kronecker product and $\mathbf{u}$ denotes the coefficient vector of U_m, that is $\mathbf{u} = \begin{pmatrix} u_{m,0} \cdots u_{m,r_m} \end{pmatrix}^T$.

By choosing the temporal shape functions to be the normalized Legendre polynomials, we get $\mathbf{G} = \mathbf{I}$ and $C_{ij} = \alpha_{ij} (i + 1/2)^{1/2} (j + 1/2)^{1/2}$, $\alpha_{ij} = (-1)^{i+j}$ if $j < i$ and 1 otherwise. The matrix $\mathbf{C}$ is diagonalizable in $\mathbb{C}$, and thus there exists a matrix $\mathbf{Q} \in \mathbb{C}^{(r+1)\times(r+1)}$ such that $\mathbf{Q}^{-1}\mathbf{C}\mathbf{Q} = \mathrm{diag}(\lambda_0, \ldots, \lambda_r)$. Multiplying (10) by $\mathbf{Q}^{-1} \otimes \mathbf{I}$ from the left gives $\left(\mathbf{T} \otimes \mathbf{M} - \frac{k}{2} \mathbf{I} \otimes \mathbf{A} \right) \mathbf{w} = \mathbf{g}$, with $\mathbf{w} = (\mathbf{Q}^{-1} \otimes \mathbf{I})\mathbf{u}$, and $\mathbf{g} = (\mathbf{Q}^{-1} \otimes \mathbf{I})\mathbf{f}$. This system is block-diagonal and completely decouples into

$$\left(\lambda_j \mathbf{M} - \frac{k}{2} \mathbf{A} \right) \mathbf{w}_j = \mathbf{g}_j, \quad j = 0, \ldots, r. \tag{11}$$

Hence, in each time-step we have to solve the $r+1$ linear systems in (11) of size N.

2.2 BDF-2

BDF-2 to solve (7) reads

$$\frac{3}{2}u_h^n = k_n A_h u_h^n + 2u_h^{n-1} - \frac{1}{2}u_h^{n-2}. \tag{12}$$

Since BDF-2 is a multi-step method we need to use a different method for the first time-step. We have used Euler-backward

$$u_h^1 = k_n A_h u_h^1 + u_h^0. \tag{13}$$

3 Solution of linear systems of equations

Both discontinuous Galerkin in time and BDF-2 leads to large systems of linear equations that have to be solved each time-step. We have solved them by performing an LU-factorization prior to the time-stepping with subsequent solves with these factors each time-step.

4 Details for different benchmark problems

The parameters that are common for all benchmark problems are:

$$\begin{aligned} d &= 0.01, \\ q &= 10 \end{aligned}$$

4.1 Benchmark problem 1–3

- The boundary conditions used for the one-dimensional problems are

$$\begin{aligned} \frac{\partial^2 u}{\partial s^2} &= 0 \quad , \quad s = s_{\min} \\ \frac{\partial^2 u}{\partial s^2} &= 0 \quad , \quad s = s_{\max} \end{aligned}$$

  together with one-sided differences for $\frac{\partial u}{\partial s}$ at both $s_{\min}$ and $s_{\max}$.

- The time-stepping method used is dG(1).

4.1.1 Problem 1

- The computation of Δ in S_0 is accomplished through a centered finite difference $\frac{\tilde{u}(S_0+\tilde{h})-\tilde{u}(S_0-\tilde{h})}{2\tilde{h}}$ where $\tilde{u}$ is an interpolation of the computed solution and $\tilde{h}$ is the smallest spatial step in the adaptive grid.

- The computation of Γ in S_0 is accomplished through a centered finite difference $\frac{\tilde{u}(S_0+\tilde{h})-2\tilde{u}(S_0)+\tilde{u}(S_0-\tilde{h})}{\tilde{h}^2}$ where $\tilde{u}$ is an interpolation of the computed solution and $\tilde{h}$ is the smallest spatial step in the adaptive grid.
- The computation of $\mathcal{V}$ in S_0 is accomplished through a centered finite difference $\frac{\tilde{u}(S_0,1.0001\sigma)-\tilde{u}(S_0,0.9999\sigma)}{0.0002\sigma}$ where $\tilde{u}$ is an interpolation of the computed solution.

Problem	$s_{\min}$	$s_{\max}$	N_c	M_c	ϵ	N_f	M_f	TM
1a) SP	0	$4K$	41	6	3.3e-3	113	6	dG(1)
1b) SP	0	$4K$	41	6	5.0e-5	989	189	BDF-2
1c) SP	0	$4K$	41	6	1.3e-3	197	11	dG(1)
1a) CP	0	$4K$	61	6	2.0e-8	61993	71	dG(1)
1b) CP	0	$4K$	61	6	3.7e-4	465	6	BDF-2
1c) CP	0	$4K$	61	6	2.0e-7	34517	69	dG(1)
1a) Δ SP	0	$4K$	41	6	8.0e-4	221	6	dG(1)
1a) Γ SP	0	$4K$	41	6	5.4e-4	269	6	dG(1)
1a) $\mathcal{V}$ SP	0	$4K$	41	6	4.1e-4	309	50	dG(1)
1a) Δ CP	0	$4K$	61	6	2.0e-8	61993	73	dG(1)
1a) Γ CP	0	$4K$	61	6	9.0e-7	92409	193	dG(1)
1a) $\mathcal{V}$ CP	0	$4K$	61	6	1.0e-8	87665	189	dG(1)

Table 1: Parameters used for Problems 1. Here SP and CP mean Standard Parameters and Challenging Parameters respectively, and TM stands for Time-stepping Method.

4.1.2 Benchmark problem 2

4.1.3 Benchmark problem 3

4.2 Benchmark problem 6

Problem	$s_{\min}$	$s_{\max}$	N_c	M_c	ϵ	N_f	M_f	TM
2) European call	0	$4K$	41 + 41	6 + 6	2.6e-4	537 + 377	6 + 6	dG(1)
2) American call	0	$4K$	41 + 41	6 + 6	2.7e-4	525 + 401	6 + 6	dG(1)

Table 2: Parameters used for Problem 2. Here TM stands for Time-stepping Method. $N_f = 537 + 377$ means that 537 spatial grid-points were used between T and αT, and 377 spatial grid-points between αT and 0, and similarily for N_c, M_f, and M_c.

Problem	$s_{\min}$	$s_{\max}$	N_c	M_c	ϵ	N_f	M_f	TM
3) Local volatility smooth	0	$4K$	41	6	3.5e-4	353	38	BDF-2
3) Local volatility implied	0	$4K$	41	6	1.7e-4	725	31	BDF-2

Table 3: Parameters used for Problem 3. TM stands for Time-stepping Method.

Problem	$s^1_{\min}$	$s^1_{\max}$	$s^2_{\min}$	$s^2_{\max}$	N^1_c	N^2_c	M_c	ϵ	N^1_f	N^2_f	M_f	TM
6)	0	350	0	175	101	101	10	3.6e-3	277	409	40	BDF-2

Table 4: Parameters used for Problem 6. TM stands for Time-stepping Method.