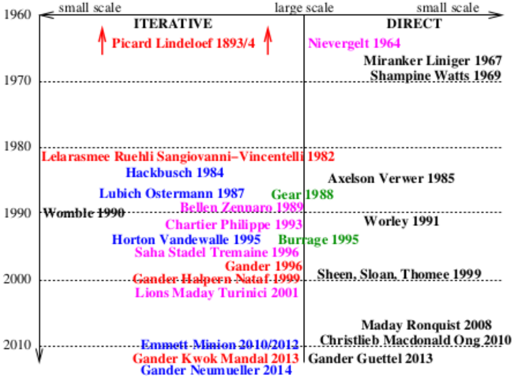
Parareal methods for optimal control I.

Julien Salomon

JLL & INRIA, ANGE project-team (Inria, Paris-Sorbonne Université, CNRS)

Ecole - Décomposition des domaines pour des problèmes de contrôle optimal,
7.9.22

Time parallelization, overview



- multiple shooting
- domain decomposition and waveform relaxation
- multigrid
- direct method
- overview papers

$\rightarrow$ *M.J. Gander, 50 Years of Time Parallel Time Integration, 2015.*

Cauchy problem on $I = [0, T]$:

$$\begin{cases} \dot{y}(t) & = & f(y(t)), \\ y(0) & = & y_0. \end{cases} \qquad (\star)$$

Decomposition: $[0, T] \to ([T_{n-1}, T_n])_{n=1,\cdots,N}$

$$(\star) \to \begin{cases} \dot{y}_n(t) & = & f(y_n(t)), \\ y_n(t_n) & = & \lambda_n. \end{cases} \qquad n = 0, \cdots, N-1.$$

Cauchy problem on $I = [0, T]$:

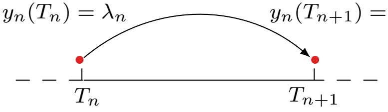
$$\begin{cases} \dot{y}(t) & = & f(y(t)), \\ y(0) & = & y_0. \end{cases} \qquad (\star)$$

Decomposition: $[0, T] \to ([T_{n-1}, T_n])_{n=1,\cdots,N}$

$$(\star) \to \begin{cases} \dot{y}_n(t) & = & f(y_n(t)), \\ y_n(t_n) & = & \lambda_n. \end{cases} \qquad n = 0, \cdots, N-1.$$

Propagator notation:

$$y_n(T_{n+1}) = P\lambda_n.$$



Reformulation: set $\Lambda = (\lambda_n)_{n=0,\cdots,N-1}$ as new unknown. It satisfies

$$\mathcal{F}(\Lambda) := \begin{pmatrix} y_0 - \lambda_0 \\ \lambda_1 - P\lambda_0 \\ \lambda_2 - P\lambda_1 \\ \vdots \\ \lambda_{N-1} - P\lambda_{N-2} \end{pmatrix} = 0.$$

Newton method:

$$\mathcal{F}'(\Lambda^k)\delta\Lambda^{k+1} = -\mathcal{F}(\Lambda^n), \text{ with } \delta\Lambda^{k+1} := \Lambda^{k+1} - \Lambda^k$$

$$\Rightarrow \delta\lambda_{n+1}^{k+1} - P'(\lambda_n^k)\ \delta\lambda_n^{k+1} = -(\lambda_{n+1}^k - P\lambda_n^k). \quad (\star\star)$$

Newton method:

$$\mathcal{F}'(\Lambda^k)\delta\Lambda^{k+1} = -\mathcal{F}(\Lambda^n),\ \text{with } \delta\Lambda^{k+1} := \Lambda^{k+1} - \Lambda^k$$

$$\Rightarrow \delta\lambda_{n+1}^{k+1} - P'(\lambda_n^k)\ \delta\lambda_n^{k+1} = -(\lambda_{n+1}^k - P\lambda_n^k). \quad (\star\star)$$

Newton method:

$$\mathcal{F}'(\Lambda^k)\delta\Lambda^{k+1} = -\mathcal{F}(\Lambda^n),\ \text{with } \delta\Lambda^{k+1} := \Lambda^{k+1} - \Lambda^k$$

$$\Rightarrow \delta\lambda_{n+1}^{k+1} - P'(\lambda_n^k)\ \delta\lambda_n^{k+1} = -(\lambda_{n+1}^k - P\lambda_n^k). \quad (\star\star)$$

Two approximations:

1. Finite differences:

$$P'(\lambda_n^k)\ \delta\lambda_n^{k+1} \approx P\lambda_n^{k+1} - P\lambda_n^k.$$

Newton method:

$$\mathcal{F}'(\Lambda^k)\delta\Lambda^{k+1} = -\mathcal{F}(\Lambda^n), \text{ with } \delta\Lambda^{k+1} := \Lambda^{k+1} - \Lambda^k$$

$$\Rightarrow \delta\lambda_{n+1}^{k+1} - P'(\lambda_n^k)\ \delta\lambda_n^{k+1} = -(\lambda_{n+1}^k - P\lambda_n^k). \quad (\star\star)$$

Two approximations:

1. Finite differences:

$$P'(\lambda_n^k)\ \delta\lambda_n^{k+1} \approx P\lambda_n^{k+1} - P\lambda_n^k.$$

2. Coarse solver:

$$P'(\lambda_n^k)\ \delta\lambda_n^{k+1} \approx P^G\lambda_n^{k+1} - P^G\lambda_n^k.$$

Newton method:

$$\mathcal{F}'(\Lambda^k)\delta\Lambda^{k+1} = -\mathcal{F}(\Lambda^n), \text{ with } \delta\Lambda^{k+1} := \Lambda^{k+1} - \Lambda^k$$

$$\Rightarrow \delta\lambda_{n+1}^{k+1} - P'(\lambda_n^k)\ \delta\lambda_n^{k+1} = -(\lambda_{n+1}^k - P\lambda_n^k). \quad (\star\star)$$

Two approximations:

1. Finite differences:

$$P'(\lambda_n^k)\ \delta\lambda_n^{k+1} \approx P\lambda_n^{k+1} - P\lambda_n^k.$$

2. Coarse solver:

$$P'(\lambda_n^k)\ \delta\lambda_n^{k+1} \approx P^G\lambda_n^{k+1} - P^G\lambda_n^k.$$

Parareal iteration:

$$(\star\star) \Rightarrow \boxed{\lambda_{n+1}^{k+1} = P\lambda_n^k + P^G\lambda_n^{k+1} - P^G\lambda_n^k}$$

Newton method:

$$\mathcal{F}'(\Lambda^k)\delta\Lambda^{k+1} = -\mathcal{F}(\Lambda^n), \text{ with } \delta\Lambda^{k+1} := \Lambda^{k+1} - \Lambda^k$$

$$\Rightarrow \delta\lambda_{n+1}^{k+1} - P'(\lambda_n^k)\ \delta\lambda_n^{k+1} = -(\lambda_{n+1}^k - P\lambda_n^k). \quad (\star\star)$$

Alternative:

$$P'(\lambda_n^k)\ \delta\lambda_n^{k+1} \approx (P^G)'(\lambda_n^{k+1} - \lambda_n^k).$$

Newton method:

$$\mathcal{F}'(\Lambda^k)\delta\Lambda^{k+1} = -\mathcal{F}(\Lambda^n),\ \text{with } \delta\Lambda^{k+1} := \Lambda^{k+1} - \Lambda^k$$

$$\Rightarrow \delta\lambda_{n+1}^{k+1} - P'(\lambda_n^k)\ \delta\lambda_n^{k+1} = -(\lambda_{n+1}^k - P\lambda_n^k). \quad (\star\star)$$

Alternative:

$$P'(\lambda_n^k)\ \delta\lambda_n^{k+1} \approx (P^G)'(\lambda_n^{k+1} - \lambda_n^k).$$

Derivative Parareal iteration:

$$(\star\star) \Rightarrow \boxed{\lambda_{n+1}^{k+1} = P\lambda_n^k + (P^G)'(\lambda_n^{k+1} - \lambda_n^k)}$$

Bibliography:

→ Original paper:

J.-L. Lions, Y. Maday, and G. Turinici. A "parareal" in time disretization of pde's. Comptes Rendus de l'Acad. des Sciences, 2001.

→ Interpretation in terms of approximate Newton's iterations:

M. Gander, S. Vandewalle, SISC Vol. 29, No. 2, pp. 556-578, 2007.

→ Derivative Parareal:

M.J. Gander and E. Hairer, Journal of Computational and Applied Mathematics, Vol. 259, pp. 1-13, 2014.

$$y(T_{n+1}) - \lambda_{n+1}^{k+1} = P y(T_n) - (P\lambda_n^k + P^G \lambda_n^{k+1} - P^G \lambda_n^k)$$

$$
\begin{aligned}
y(T_{n+1}) - \lambda_{n+1}^{k+1} =& P y(T_n) - (P\lambda_n^k + P^G \lambda_n^{k+1} - P^G \lambda_n^k) \\
=& P y(T_n) - (P\lambda_n^k + P^G \lambda_n^{k+1} - P^G \lambda_n^k) + (-P^G y(T_n) + P^G y(T_n))
\end{aligned}
$$

$$\begin{aligned}
y(T_{n+1}) - \lambda_{n+1}^{k+1} =& P y(T_n) - (P\lambda_n^k + P^G \lambda_n^{k+1} - P^G \lambda_n^k) \\
=& P y(T_n) - (P\lambda_n^k + P^G \lambda_n^{k+1} - P^G \lambda_n^k) + (-P^G y(T_n) + P^G y(T_n)) \\
=& P y(T_n) - P\lambda_n^k - P^G \lambda_n^{k+1} + P^G \lambda_n^k - P^G y(T_n) + P^G y(T_n)
\end{aligned}$$

$$\begin{aligned}
y(T_{n+1}) - \lambda_{n+1}^{k+1} =& P y(T_n) - (P\lambda_n^k + P^G\lambda_n^{k+1} - P^G\lambda_n^k) \\
=& P y(T_n) - (P\lambda_n^k + P^G\lambda_n^{k+1} - P^G\lambda_n^k) + (-P^G y(T_n) + P^G y(T_n)) \\
=& P y(T_n) - P\lambda_n^k - P^G\lambda_n^{k+1} + P^G\lambda_n^k - P^G y(T_n) + P^G y(T_n) \\
=& (P - P^G) y(T_n) - (P - P^G)\lambda_n^k + P^G y(T_n) - P^G\lambda_n^{k+1}.
\end{aligned}$$

$$
\begin{aligned}
y(T_{n+1}) - \lambda_{n+1}^{k+1} =& P y(T_n) - (P\lambda_n^k + P^G\lambda_n^{k+1} - P^G\lambda_n^k) \\
=& P y(T_n) - (P\lambda_n^k + P^G\lambda_n^{k+1} - P^G\lambda_n^k) + (-P^G y(T_n) + P^G y(T_n)) \\
=& P y(T_n) - P\lambda_n^k - P^G\lambda_n^{k+1} + P^G\lambda_n^k - P^G y(T_n) + P^G y(T_n) \\
=& (P - P^G) y(T_n) - (P - P^G)\lambda_n^k + P^G y(T_n) - P^G\lambda_n^{k+1}.
\end{aligned}
$$

Assumptions:

- Truncation error: $\|(P - P^G)x\| \leq C_1 \Delta T^{p+1} \|x\|$,
- Lipschtiz property for P^G: $\|P^G x - P^G y\| \leq (1 + C_2 \Delta T)\|x - y\|$.

$$\begin{aligned}
y(T_{n+1}) - \lambda_{n+1}^{k+1} =& P y(T_n) - (P\lambda_n^k + P^G \lambda_n^{k+1} - P^G \lambda_n^k) \\
=& P y(T_n) - (P\lambda_n^k + P^G \lambda_n^{k+1} - P^G \lambda_n^k) + (-P^G y(T_n) + P^G y(T_n)) \\
=& P y(T_n) - P\lambda_n^k - P^G \lambda_n^{k+1} + P^G \lambda_n^k - P^G y(T_n) + P^G y(T_n) \\
=& (P - P^G) y(T_n) - (P - P^G)\lambda_n^k + P^G y(T_n) - P^G \lambda_n^{k+1}.
\end{aligned}$$

Assumptions:

- Truncation error: $\|(P - P^G)x\| \leq C_1 \Delta T^{p+1} \|x\|$,
- Lipschtiz property for P^G: $\|P^G x - P^G y\| \leq (1 + C_2 \Delta T)\|x - y\|$.

$$\|y(T_{n+1}) - \lambda_{n+1}^{k+1}\| \leq C_1 \Delta T^{p+1} \|y(T_n) - \lambda_n^k\| + (1 + C_2 \Delta T)\|y(T_n) - \lambda_n^{k+1}\|.$$

$$
\begin{aligned}
y(T_{n+1}) - \lambda_{n+1}^{k+1} =& P y(T_n) - (P\lambda_n^k + P^G\lambda_n^{k+1} - P^G\lambda_n^k) \\
=& P y(T_n) - (P\lambda_n^k + P^G\lambda_n^{k+1} - P^G\lambda_n^k) + (-P^G y(T_n) + P^G y(T_n)) \\
=& P y(T_n) - P\lambda_n^k - P^G\lambda_n^{k+1} + P^G\lambda_n^k - P^G y(T_n) + P^G y(T_n) \\
=& (P - P^G) y(T_n) - (P - P^G)\lambda_n^k + P^G y(T_n) - P^G\lambda_n^{k+1}.
\end{aligned}
$$

Assumptions:

- Truncation error: $\|(P - P^G)x\| \leq C_1 \Delta T^{p+1} \|x\|$,
- Lipschtiz property for P^G: $\|P^G x - P^G y\| \leq (1 + C_2\Delta T)\|x - y\|$.

$$
\|y(T_{n+1}) - \lambda_{n+1}^{k+1}\| \leq C_1 \Delta T^{p+1} \|y(T_n) - \lambda_n^k\| + (1 + C_2\Delta T)\|y(T_n) - \lambda_n^{k+1}\|.
$$

$$\|y(T_{n+1}) - \lambda_{n+1}^{k+1}\| \leq C_1 \Delta T^{p+1} \|y(T_n) - \lambda_n^k\| + (1 + C_2 \Delta T) \|y(T_n) - \lambda_n^{k+1}\|.$$

$$\Downarrow$$

$$\begin{aligned}\|y(T_n) - \lambda_n^k\| \leq & \frac{(C_1 \Delta T^{p+1})^{k+1}}{(k+1)!} (1 + C_2 \Delta T)^{n-k-1} \Pi_{j=0}^{k} (n-j) \\ \leq & \frac{(C_1 T_n)^{k+1}}{(k+1)!} e^{C_2 (T_n - T_{k+1})} \Delta T^{p(k+1)}.\end{aligned}$$

$\rightarrow$ *Detailed analysis:*

M.J. Gander and E. Hairer, Nonlinear Convergence Analysis for the Parareal Algorithm, 2007.

Outline

General stiff Vlasov equation:

$$\frac{\partial f_\varepsilon}{\partial t} + \frac{1}{\varepsilon} v \frac{\partial f_\varepsilon}{\partial r} + E_\varepsilon \frac{\partial f_\varepsilon}{\partial v} = 0, \quad f_\varepsilon(t=0,r,v) = f_0(r,v),$$

where

- $f_\varepsilon = f_\varepsilon(t,r,v)$ is the distribution function of a particle species,
- $r, v \in \mathbb{R}$ are the position and velocity,
- $E_\varepsilon = E_\varepsilon(t,r) \in \mathbb{R}$ plays the role of the electric forces.

$\rightarrow$ *Details to appear in:*

L. Grigori, S.A. Hirstoaga, J. Salomon, A Parareal algorithm for some Vlasov-Poisson equation with reduced model for the coarse solving, 2022.

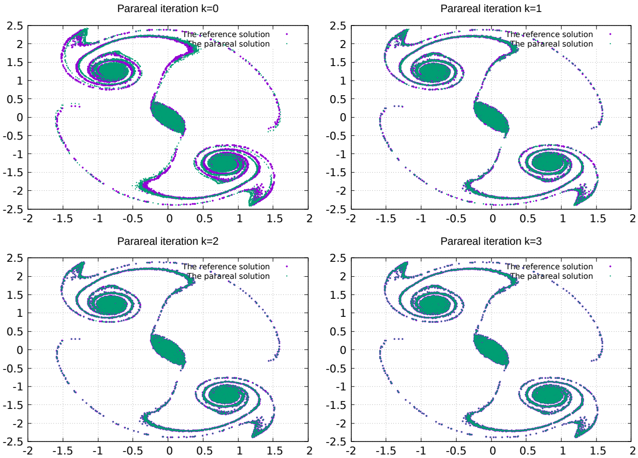


Figure: The parareal solution of the Vlasov-Poisson equation, final time $T = 36$.

"Non-linear control" or "Bilinear control"

	Linear eq.	Non-linear eq.
"Linear" control	$\dot{y} = Ay + Bc$	$\dot{y} = f(y) + Bc$
Non-linear control	$\dot{y} = A(c)y$	$\dot{y} = f(y, c)$

- $y = y(t, x)$ state
- $c = c(t)$ or $c(t, x)$ control

Outline

Problem: control on a fixed, bounded interval $[0, T]$

Given $T > 0$, consider the optimal control problem associated with the cost functional

$$J(c) = \frac{1}{2}\|y(T) - y_{target}\|^2 + \frac{\alpha}{2}\int_0^T c^2(t)dt,$$

where the state function x evolution is described by an equation:

$$\dot{y}(t) = f(y(t), c(t)),$$

with initial condition $y(0) = y_{init}$.

Objective:	Given an optimal control solver, combine it with a time-parallelization.

"Non-linear control" or "Bilinear control"

	Linear eq.	Non-linear eq.
"Linear" control	$\dot{y} = Ay + Bc$	$\dot{y} = f(y) + Bc$
Non-linear control	$\dot{y} = A(c)y$	$\dot{y} = f(y, c)$

- $y = y(t, x)$ state
- $c = c(t)$ or $c(t, x)$ control

Optimality system (1/2)

How to characterize the optima ?

$\rightarrow$ Solve the *Euler-Lagrange* equations.

- Define the *Lagrange operator*:

$$\mathcal{L}(y, \lambda, c) = J(c) - \int_0^T \lambda(t) \cdot (\dot{y}(t) - f(y(t), c(t)))dt$$

- Compute its partial derivatives.
- Cancel them !

Optimality system (2/2)

The optima are characterized by the Euler-Lagrange equations:

$$\begin{aligned}
\dot{y}(t) =& f(y(t), c(t)) \\
y(t=0) =& y_0 \\
\dot{\lambda}(t) =& -[\partial_y f(y(t), c(t))]\lambda(t) \\
\lambda(t=T) =& y(T) - y_{target} \\
\alpha c(t) =& -\lambda(t) \cdot \partial_c f(y(t), c(t))
\end{aligned}$$

"Linear control"

	Linear eq.	Non-linear eq.
"Linear" control	$\dot{y} = Ay + Bc$	$\dot{y} = f(y) + Bc$
Non-linear control	$\dot{y} = A(c)y$	$\dot{y} = f(y, c)$

- $y = y(t, x)$ state
- $c = c(t)$ or $c(t, x)$ control

The optimality condition then reads

$$\left\{\begin{array}{rcl} \dot{y}(t) & = & f(y(t)) + c(t), \\ \dot{\lambda}(t) & = & -(f(y(t))')^T \lambda(t), \\ \alpha c(t) & = & -\lambda(t). \end{array}\right.$$

$\rightarrow$ **Elimination of c:**

$$\left\{\begin{array}{l} \dot{y} = f(y) - \dfrac{\lambda}{\alpha}, \\ \\ \dot{\lambda} = -(f(y)')^T \lambda, \end{array}\right.$$

and final condition $\lambda(T) = y(T) - y_{target}$.

Boundary value problems notations : on the subinterval $[T_l, T_{l+1}]$ with initial condition $y(T_l) = y_l$ and final condition $\lambda(T_{l+1}) = \lambda_{l+1}$, we denote

$$\begin{pmatrix} y(T_{l+1}) \\ \lambda(T_l) \end{pmatrix} = \begin{pmatrix} P(y_l, \lambda_{l+1}) \\ Q(y_l, \lambda_{l+1}) \end{pmatrix}.$$

The optimality system is enriched:

$$
\begin{array}{rclrcl}
y_0 - y_{init} & = & 0 & & & \\
y_1 - P(y_0, \lambda_1) & = & 0 & \lambda_1 - Q(y_1, \lambda_2) & = & 0 \\
y_2 - P(y_1, \lambda_2) & = & 0 & \lambda_2 - Q(y_2, \lambda_3) & = & 0 \\
& \vdots & & & \vdots & \\
y_L - P(y_{L-1}, \lambda_L) & = & 0 & \lambda_L - y_L + y_{target} & = & 0
\end{array}
\tag{1}
$$

That is : **a system of boundary value subproblems, satisfying matching conditions**.

Collecting the unknowns in the vector

$$(Y^T, \Lambda^T) := (y_0, y_1, y_2, \ldots, y_L, \lambda_1, \lambda_2, \ldots, \lambda_L),$$

we obtain the nonlinear system

$$\mathcal{F}(Y, \Lambda) := \begin{pmatrix} y_0 - y_{init} \\ y_1 - P(y_0, \lambda_1) \\ y_2 - P(y_1, \lambda_2) \\ \vdots \\ y_L - P(y_{L-1}, \lambda_L) \\ \lambda_1 - Q(y_1, \lambda_2) \\ \lambda_2 - Q(y_2, \lambda_3) \\ \vdots \\ \lambda_L - y_L + y_{target} \end{pmatrix} = 0.$$

Newton's method:

$$\mathcal{F}'\begin{pmatrix} Y^n \\ \Lambda^n \end{pmatrix}\begin{pmatrix} Y^{n+1}-Y^n \\ \Lambda^{n+1}-\Lambda^n \end{pmatrix} = -\mathcal{F}\begin{pmatrix} Y^n \\ \Lambda^n \end{pmatrix},$$

where the Jacobian matrix of $\mathcal{F}$ is given by

$$\mathcal{F}'\begin{pmatrix} Y \\ \Lambda \end{pmatrix} =$$

$$\left(\begin{array}{cccc|cccc}
1 & & & & & & & \\
-P_Y(Y_0,\Lambda_1) & 1 & & & -P_\Lambda(Y_0,\Lambda_1) & & & \\
 & \ddots & \ddots & & & \ddots & & \\
 & & -P_Y(Y_{N-1},\Lambda_N) & 1 & & & & -P_\Lambda(Y_{N-1},\Lambda_N) \\
\hline
 & -Q_Y(Y_1,\Lambda_2) & & & 1 & -Q_\Lambda(Y_1,\Lambda_2) & & \\
 & \ddots & & & & \ddots & \ddots & \\
 & & -Q_Y(Y_{N-1},\Lambda_N) & & & & 1 & -Q_\Lambda(Y_{N-1},\Lambda_N) \\
 & & & -1 & & & & 1
\end{array}\right)$$

Following Parareal strategy, we get:

$$
\begin{aligned}
P_y(y_{\ell-1}^n,\lambda_\ell^n)(y_{\ell-1}^{n+1}-y_{\ell-1}^n) &\approx P^G(y_{\ell-1}^{n+1},\lambda_\ell^n)-P^G(y_{\ell-1}^n,\lambda_\ell^n),\\
P_\lambda(y_{\ell-1}^n,\lambda_\ell^n)(\lambda_\ell^{n+1}-\lambda_\ell^n) &\approx P^G(y_{\ell-1}^n,\lambda_\ell^{n+1})-P^G(y_{\ell-1}^n,\lambda_\ell^n),\\
Q_\lambda(y_{\ell-1}^n,\lambda_\ell^n)(\lambda_\ell^{n+1}-\lambda_\ell^n) &\approx Q^G(y_{\ell-1}^n,\lambda_\ell^{n+1})-Q^G(y_{\ell-1}^n,\lambda_\ell^n),\\
Q_y(y_{\ell-1}^n,\lambda_\ell^n)(y_{\ell-1}^{n+1}-y_{\ell-1}^n) &\approx Q^G(y_{\ell-1}^{n+1},\lambda_\ell^n)-Q^G(y_{\ell-1}^n,\lambda_\ell^n).
\end{aligned}
$$

As usual for Parareal-type approaches:

- In parallel: all fine propagations on sub-intervals.
- Sequential part: only coarse solving.

Or, alternatively, with the derivative parareal approach:

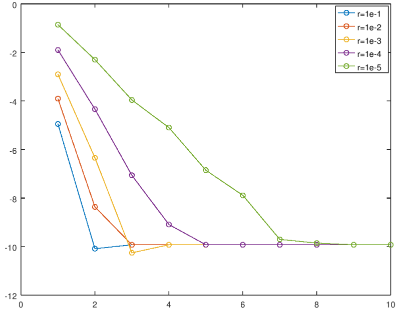
$$
\begin{array}{rcl}
P_y(y^n_{\ell-1},\lambda^n_\ell)(y^{n+1}_{\ell-1}-y^n_{\ell-1}) & \approx & (P^G_y)'(y^n_{\ell-1},\lambda^n_\ell)(y^{n+1}_{\ell-1}-y^n_{\ell-1}),\\
P_\lambda(y^n_{\ell-1},\lambda^n_\ell)(\lambda^{n+1}_\ell-\lambda^n_\ell) & \approx & (P^G_\lambda)'(y^n_{\ell-1},\lambda^n_\ell)(\lambda^{n+1}_\ell-\lambda^n_\ell),\\
Q_\lambda(y^n_{\ell-1},\lambda^n_\ell)(\lambda^{n+1}_\ell-\lambda^n_\ell) & \approx & (Q^G_\lambda)'(y^n_{\ell-1},\lambda^n_\ell)(\lambda^{n+1}_\ell-\lambda^n_\ell),\\
Q_y(y^n_{\ell-1},\lambda^n_\ell)(y^{n+1}_{\ell-1}-y^n_{\ell-1}) & \approx & (Q^G_y)'(y^n_{\ell-1},\lambda^n_\ell)(y^{n+1}_{\ell-1}-y^n_{\ell-1}).
\end{array}
$$

As usual for Parareal-type approaches:

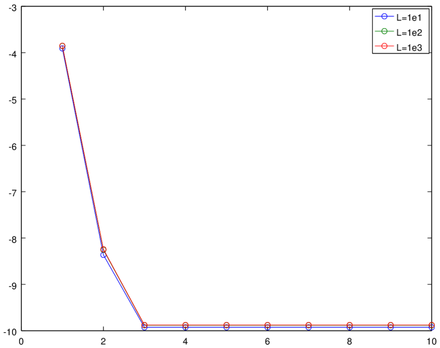
- In parallel: all fine propagations on sub-intervals.
- Sequential part: only coarse solving.

Example : 1D, linear dynamics, Dahlquist system

$$\dot{y}(t) = \sigma y(t) + c(t).$$



Error vs. # iterations, various $r = \delta t / \Delta t$, fixed $\delta t = \delta t_0$.



Error vs. # iterations, various number of subintervals.

Example : 2D, nonlinear dynamics, Lotka-Volterra system

- Minimize

$$J(c) = \frac{1}{2}|y(1) - y_{\mathsf{target}}|^2 + \frac{1}{2}\int_0^1 |c(t)|^2 \, dt$$

with $y_{\mathsf{target}} = (100, 20)^T$, subject to the Lotka-Volterra equation

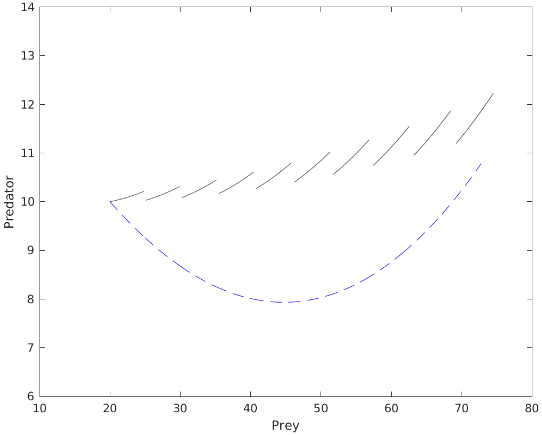
$$\dot{y}_1 = a_1 y_1 - b_1 y_1 y_2 + c_1, \quad \dot{y}_2 = a_2 y_1 y_2 - b_2 y_2 + c_2$$

with initial conditions $y(0) = (20, 10)^T$

- Backward Euler, $\delta t = 10^{-5}$

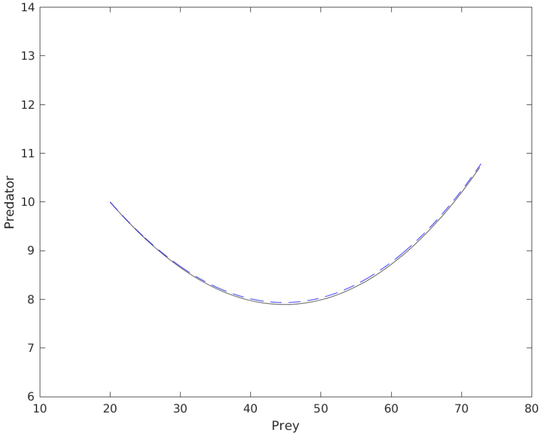
Vector example - $N = 10$, $r = \delta t / \Delta t = 0.01$

Iteration #1



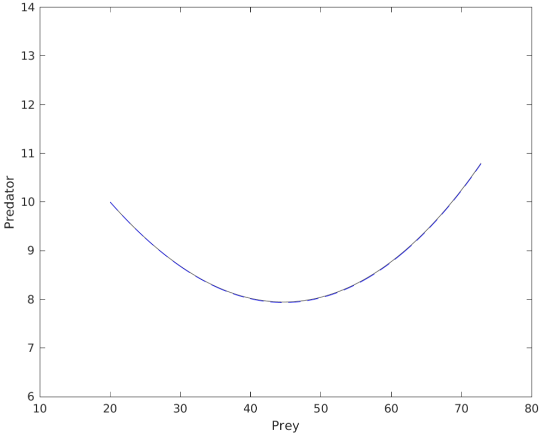
Vector example - $N = 10$, $r = \delta t/\Delta t = 0.01$

Iteration #2



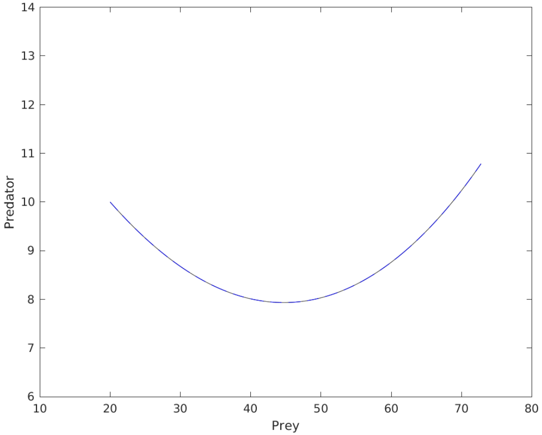
Vector example - $N = 10$, $r = \delta t/\Delta t = 0.01$

Iteration #3



Vector example - $N = 10$, $r = \delta t / \Delta t = 0.01$

Iteration #4



- Minimize

$$J(c) = \frac{1}{2}|y(20) - y_{\mathsf{target}}|^2 + \frac{1}{2}\int_0^{20} |c(t)|^2 \, dt$$

with $y_{\mathsf{target}} = (100, 20)^T$, subject to the Lotka-Volterra equation

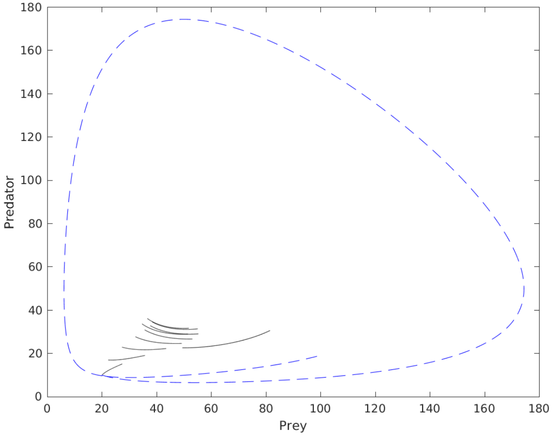
$$\dot{y}_1 = a_1 y_1 - b_1 y_1 y_2 + c_1, \quad \dot{y}_2 = a_2 y_1 y_2 - b_2 y_2 + c_2$$

with initial conditions $y(0) = (20, 10)^T$

- Backward Euler, $\delta t = 20 \cdot 10^{-5}$

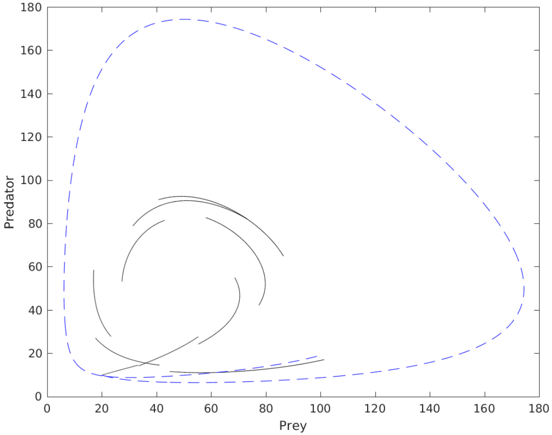
Vector example - $N = 10$, $r = \delta t / \Delta t = 0.01$

Iteration #1



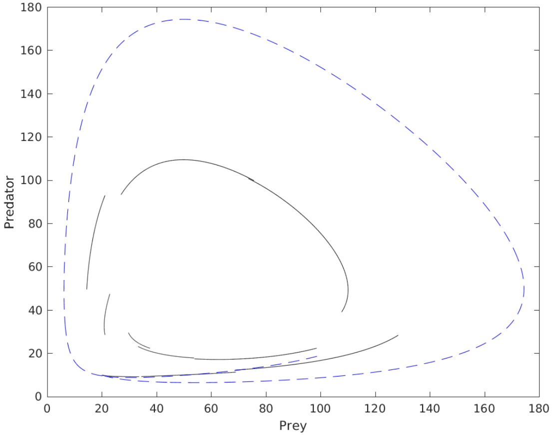
Vector example - $N = 10$, $r = \delta t / \Delta t = 0.01$

Iteration #2



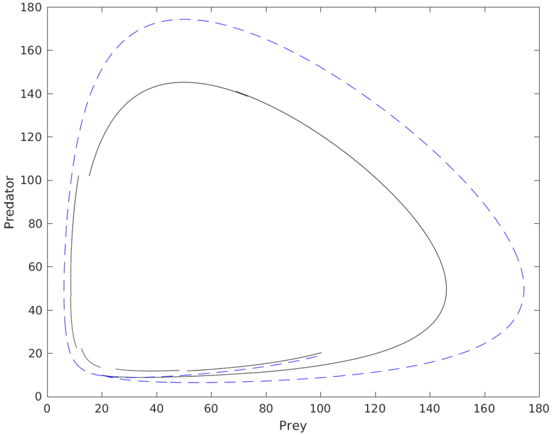
Vector example - $N = 10$, $r = \delta t/\Delta t = 0.01$

Iteration #3



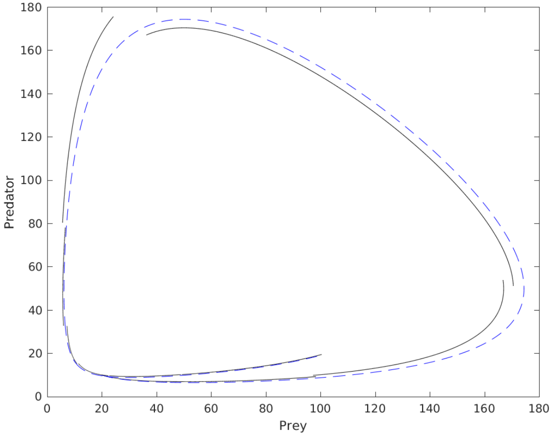
Vector example - $N = 10$, $r = \delta t/\Delta t = 0.01$

Iteration #4



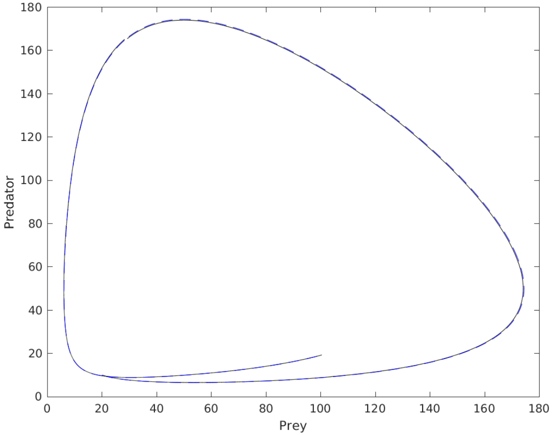
Vector example - $N = 10$, $r = \delta t / \Delta t = 0.01$

Iteration #5



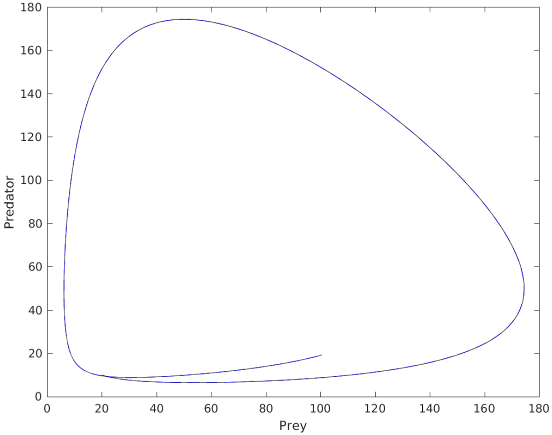
Vector example - $N = 10$, $r = \delta t / \Delta t = 0.01$

Iteration #6



Vector example - $N = 10$, $r = \delta t/\Delta t = 0.01$

Iteration #7



Trick : Derivative Evaluation by **Gauss-Newton**

- Approximation: neglect 2nd derivatives

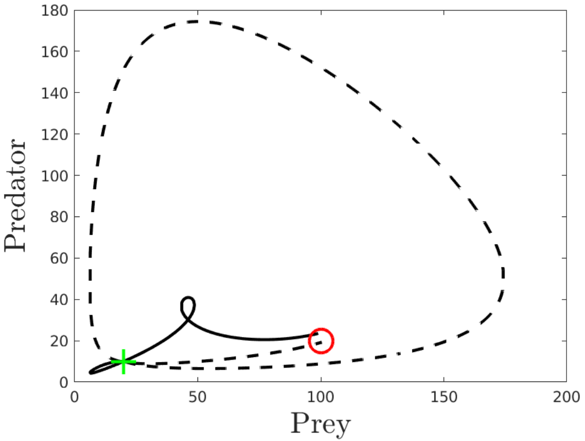
$$\frac{dy'}{dt} = f'(y)y' - \frac{\lambda'}{\alpha}, \qquad y'(0) = Y_n^{k+1} - Y_n^k,$$

$$\frac{d\lambda'}{dt} = -(f'(y))^T\lambda' - \cancel{(f''(y,y'))^T\lambda}, \qquad \lambda'(T) = \Lambda_{n+1}^{k+1} - \Lambda_{n+1}^k.$$

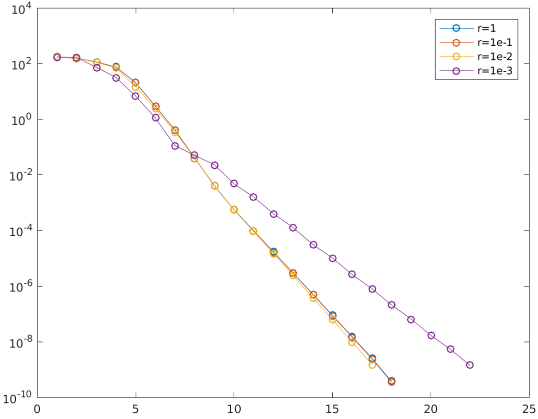
- Simplified ODE for λ' independent of y'
- Approximate derivatives in one backward-forward sweep!

Vector example - $N = 10$, $r = \delta t / \Delta t = 0.01$

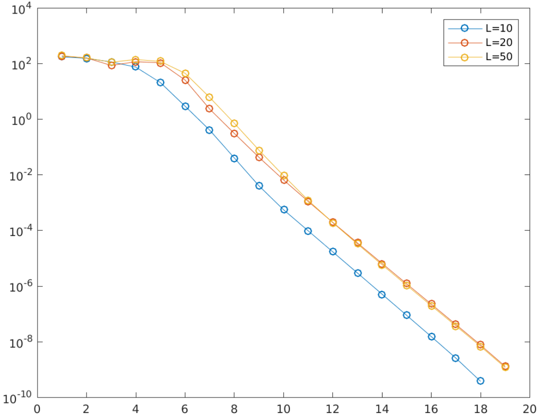
Iteration #7



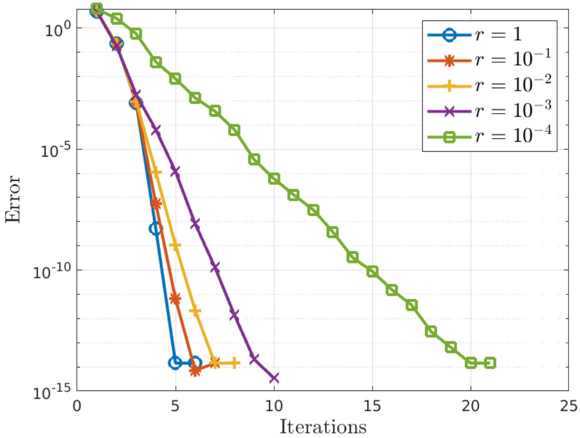
$N = 10$ subdomains, varying $r = \delta t/\Delta t$



$\delta t/\Delta t = 0.01$, varying # subdomains



True Newton:


10^{0}
10^{-5}
10^{-10}
10^{-15}
0
5
10
15
20
25
Iterations
Error
$r = 1$
$r = 10^{-1}$
$r = 10^{-2}$
$r = 10^{-3}$
$r = 10^{-4}$

Consider, after space discretization, the dynamic:

$$\dot{y}(t) = Ay(t) + c(t),$$

where A is a real, symmetric matrix with **negative** eigenvalues.

$\rightarrow$ **Example:** A obtained with a finite difference discretization of a diffusion operator in space.

We use the *Discretize-then-optimize* strategy:

$$\dot{y}(t) = Ay(t) + c(t) \rightarrow y_{n+1} = y_n + \delta t(Ay_{n+1} + c_{n+1}).$$

We use the *Discretize-then-optimize* strategy:

$$\dot{y}(t) = Ay(t) + c(t) \to y_{n+1} = y_n + \delta t(Ay_{n+1} + c_{n+1}).$$

And the discrete cost functional is:

$$J_{\delta t}(c) = \frac{1}{2}\|y_M - y_{target}\|^2 + \frac{\alpha}{2}\delta t \sum_{n=0}^{M-1} \|c_{n+1}\|^2.$$

We use the *Discretize-then-optimize* strategy:

$$\dot{y}(t) = Ay(t) + c(t) \to y_{n+1} = y_n + \delta t(Ay_{n+1} + c_{n+1}).$$

And the discrete cost functional is:

$$J_{\delta t}(c) = \frac{1}{2}\|y_M - y_{target}\|^2 + \frac{\alpha}{2}\delta t \sum_{n=0}^{M-1} \|c_{n+1}\|^2.$$

And the discrete optimality system is given by:

$$\begin{aligned}
y_0 &= y_{init}, && \\
y_{n+1} &= (I - \delta t A)^{-1}(y_n + \delta t c_{n+1}), && n = 0, 1, \dots, M-1, \\
\lambda_M &= y_M - y_{target}, && \\
\lambda_n &= (I - \delta t A)^{-1}\lambda_{n+1}, && n = 0, 1, \dots, M-1, \\
\alpha c_{n+1} &= -(I - \delta t A)^{-1}\lambda_{n+1}, && n = 0, 1, \dots, M-1,
\end{aligned}$$

By diagonalization (since A is real symetric):

$$y_{n+1} = y_n + \delta t(Ay_{n+1} + c_{n+1}) \rightarrow y_{n+1} = y_n + \delta t(\sigma y_{n+1} + c_{n+1}),$$

$\Rightarrow$ **Reduction to independent Dahlquist systems.**

By diagonalization (since A is real symetric):

$$y_{n+1} = y_n + \delta t(Ay_{n+1} + c_{n+1}) \rightarrow y_{n+1} = y_n + \delta t(\sigma y_{n+1} + c_{n+1}),$$

$\Rightarrow$ **Reduction to independent Dahlquist systems.**
And we get formulas like:

$$\begin{aligned} y_{n_2} &= (1-\sigma\delta t)^{n_1-n_2} y_{n_1} - \delta t \sum_{j=0}^{n_2-n_1-1} (1-\sigma\delta t)^{n_1-n_2+j} c_{n_1+j+1} \\ &= (1-\sigma\delta t)^{n_1-n_2} y_{n_1} - \frac{\delta t}{\alpha} \sum_{j=0}^{n_2-n_1-1} (1-\sigma\delta t)^{n_1-n_2+j-1} \lambda_{n_1+j+1}. \end{aligned}$$

So that we end up with the reduced system:

$$\begin{aligned} Y_0 &= y_{init} & \\ -\beta_{\delta t} Y_{\ell-1} + Y_\ell + \frac{\gamma_{\delta t}}{\alpha}\Lambda_\ell &= 0, & 1 \le \ell \le M, \\ \Lambda_{\ell-1} - \beta_{\delta t}\Lambda_\ell &= 0, & 0 \le \ell \le M-1, \\ Y_\ell + \Lambda_\ell &= y_{target}, & \end{aligned}$$

where

$$\begin{aligned} \beta_{\delta t} &:= (1-\sigma\delta t)^{-\Delta T/\delta t}, \\ \gamma_{\delta t} &:= \delta t \sum_{j=0}^{N-1} (1-\sigma\delta t)^{2(j-N)} = \frac{\beta_{\delta t}^2 - 1}{\sigma(2-\sigma\delta t)}. \end{aligned}$$

In a matrix form

$$\Rightarrow \left(\begin{array}{cccc|cccc}
1 & & & & 0 & & & \\
-\beta_{\delta t} & \ddots & & & \gamma_{\delta t}/\alpha & \ddots & & \\
 & \ddots & \ddots & & & \ddots & & 0 \\
 & & -\beta_{\delta t} & 1 & & & & \gamma_{\delta t}/\alpha \\
\hline
 & & & & 1 & -\beta_{\delta t} & & \\
 & & & & & \ddots & \ddots & \\
 & & & & & & \ddots & -\beta_{\delta t} \\
 & & & -1 & & & & 1
\end{array}\right)
\left(\begin{array}{c}
Y_0 \\ \vdots \\ \vdots \\ Y_L \\ \hline \Lambda_1 \\ \vdots \\ \vdots \\ \Lambda_L
\end{array}\right)
=
\left(\begin{array}{c}
y_{init} \\ 0 \\ \\ \vdots \\ \\ 0 \\ -y_{target}
\end{array}\right).$$

or, in a more compact form,

$$M_{\delta t} X = b.$$

Conclusion: we need to characterize the eigenvalues of $Id - M_{\Delta t}^{-1} M_{\delta t}$.

$$\Rightarrow (M_{\Delta t} - M_{\delta t})x = \mu M_{\Delta t} x,$$

i.e., after some computation, study the roots of:

$$P(\mu) = \alpha \mu^{2L-1} + (\mu\gamma - \delta\gamma) \sum_{\ell=0}^{L-1} \mu^{2(L-\ell-1)} (\mu\beta - \delta\beta)^{2\ell},$$

where

$$\beta = \beta_{\Delta t},\ \gamma = \gamma_{\Delta t},\ \delta\beta = \beta_{\Delta t} - \beta_{\delta t},\ \delta\gamma = \gamma_{\Delta t} - \gamma_{\delta t}.$$

Theorem

Let ΔT, Δt, δt and α be fixed. Then for all $\sigma < 0$, the spectral radius of $I - M_{\Delta t}^{-1} M_{\delta t}$ satisfies

$$\max_{\sigma<0} \rho(\sigma) \leq \frac{0.79\Delta t}{\alpha + \sqrt{\alpha \Delta t}} + 0.3.$$

Thus, if $\alpha > 0.4544\Delta t$, then the linear ParaOpt algorithm converges.

F. Kwok, M. Gander, J. Salomon,

SISC Vol. 42, No. 5, pp. A2773–A2802.

Numerical test:

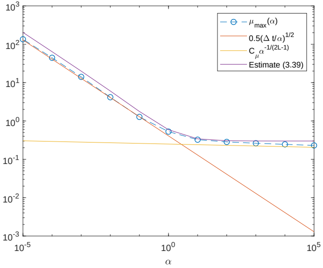


Figure: Behaviour of $\max_{\sigma<0} \rho(\sigma)$ as a function of α, $T = 100$, $L = 30$, $\Delta T = \Delta t$, $\Delta t/\delta t = 10^{-4}$. The data for $\mu_{\max}(\alpha)$ has been generated by solving the generalized eigenvalue problem using `eig` in MATLAB.

Ingredients of the proof:

- Analysis of the functions β, γ, ...
- Complex analysis: Argument Principle.

Setting: Dahlquist equation, $\sigma > 0$, explicit Euler discretization

$$y_{n+1} = y_n + \delta t(\sigma y_n + c_n).$$

Reduced to

$$P(Y_\ell, \Lambda_{\ell+1}) := \beta Y_\ell - \frac{\gamma}{\alpha}\Lambda_{\ell+1},$$
$$Q(Y_\ell, \Lambda_{\ell+1}) := \beta \Lambda_{\ell+1},$$

with

$$\beta := (1 + \sigma\delta t)^{\frac{\Delta T}{\delta t}},$$

$$\gamma_{\delta t} := \frac{\beta^2 - 1}{\sigma(2 + \sigma\delta t)}.$$

- P has only one root τ_0 in $]1, \infty[$.
- Let $\tau = \beta - \gamma\delta\beta/\delta\gamma$ and L_0 be

$$L_0 = \frac{(\beta - \tau)}{\gamma(\tau - \tau_0)}.$$

Theorem

Let $\sigma > 0, \alpha, T, L, \Delta t, \delta t$ and be fixed. If $P(\tau) > 0$ and L satisfies $L > \alpha L_0$ then, the spectral radius of $\left(Id - M_{\Delta t}^{-1} M_{\delta t}\right)$ satisfies

$$\rho < (\Delta t - \delta t)\, C(\sigma, \Delta T),$$

where

$$C(\sigma, \Delta T) = \sigma \left[\frac{1}{2} + \left(\frac{1}{2}\sigma \Delta T + 1\right) e^{2\sigma \Delta T}\right].$$

Ingredients of the proof:

- Analysis of the functions β, γ, ...
- Complex analysis: Rouché's Theorem.

N. Tognon, to appear.

Partial end...