Supervisor: Prof. Manuel Torrilhon

Comparison of Time Stepping Techniques for Compressible Gas Dynamics

Julian Köllermeier

RWTH Aachen University

May 3rd, 2012

Outline

Introduction

Aim of the Project

Implementation and comparison
of efficient implicit time stepping schemes
for non-linear PDE systems

Aim of the Project

Implementation and comparison
of efficient implicit time stepping schemes
for non-linear PDE systems

Steps

- implicit time discretization using adaptive time stepping
- non-linear solver and computation of Jacobian
- preconditioned linear solver
- comparison of methods

Aim of the Project

Implementation and comparison
of efficient implicit time stepping schemes
for non-linear PDE systems

Steps

- implicit time discretization using adaptive time stepping
- non-linear solver and computation of Jacobian
- preconditioned linear solver
- comparison of methods

$\Rightarrow$ runtime speed-up

Example: Heat Equation

$$\frac{\partial T}{\partial t} + \nabla \underbrace{\left(-\kappa \nabla T\right)}_{q} = 0$$

Example: Heat Equation

$$\frac{\partial T}{\partial t} + \nabla \underbrace{(-\kappa \nabla T)}_{q} = 0$$

Fourier's law

heat flux q proportional to negative temperature gradient over surface

$$q = -\kappa(T)\nabla T$$

Example: Heat Equation

$$\frac{\partial T}{\partial t} + \nabla \underbrace{(-\kappa \nabla T)}_{q} = 0$$

Fourier's law

heat flux q proportional to negative temperature gradient over surface

$$q = -\kappa(T)\nabla T$$

Models for heat conductivity κ

- $\kappa = 1 = const$ (linear)
- $\kappa(T) = c_1 + c_2 \cdot T^2$ (non-linear), $c_1, c_2 \in \mathbb{R}^+$

Explicit vs. Implicit schemes

Explicit schemes

+ implementation
+ no linear or non-linear solver needed
– CFL condition

Explicit vs. Implicit schemes

Explicit schemes

+ implementation
+ no linear or non-linear solver needed
– CFL condition

Implicit schemes

– implementation
– solution of large non-linear system of equations
– needs Jacobian
+ no time step constraints

Numerics

Spatial Discretization

$$\frac{\partial Q}{\partial t} + L(Q) = 0$$

Spatial Discretization

$$\frac{\partial Q}{\partial t} + L(Q) = 0$$

Spatial discretization scheme by KAPPER

- least-squares reconstruction of fluxes
- second order in space
- six-point stencil for flux reconstruction
- nine-point stencil for each cell

Spatial Discretization

$$\frac{\partial Q}{\partial t} + L(Q) = 0$$

Spatial discretization scheme by KAPPER

- least-squares reconstruction of fluxes
- second order in space
- six-point stencil for flux reconstruction
- nine-point stencil for each cell

$$\frac{\partial Q}{\partial t} + R(Q) = 0$$

Time Discretization

Apply different time stepping methods to

$$\frac{\partial Q}{\partial t} + R(Q) = 0$$

Time Discretization

Apply different time stepping methods to

$$\frac{\partial Q}{\partial t} + R(Q) = 0$$

Time stepping methods

- implicit Euler method
- implicit midpoint method
- implicit trapezoidal method
- BDF2 method
- Richardson extrapolation

Adaptive Time Stepping

Aim

calculate with largest possible time step Δt subject to given error bound ϵ and stability of the method

Adaptive Time Stepping

Aim

calculate with largest possible time step Δt subject to given error bound ϵ and stability of the method

Ingredients

- sensor for the error (error estimate)
- controller for Δt or h (time step adjustment strategy)
- use control theory model

Error Estimation

Comparison with higher order method

$$\widehat{r}_{n+1} = \|y_1 - y_2\|$$

Error Estimation

Comparison with higher order method

$$\widehat{r}_{n+1} = \|y_1 - y_2\|$$

Comparison by step size variation

$$y^{h}_{n+1} = y(t_{n+1}) + \phi(t_n)h^k + \mathcal{O}(h^{k+1})$$

$$y^{\frac{h}{m}}_{n+1} = y(t_{n+1}) + \phi(t_n)\frac{h}{m}^k + \mathcal{O}\left(\frac{h}{m}^{k+1}\right)$$

Error Estimation

Comparison with higher order method

$$\widehat{r}_{n+1} = \|y_1 - y_2\|$$

Comparison by step size variation

$$y^h_{n+1} = y(t_{n+1}) + \phi(t_n)h^k + \mathcal{O}(h^{k+1})$$

$$y^{\frac{h}{m}}_{n+1} = y(t_{n+1}) + \phi(t_n)\frac{h}{m}^k + \mathcal{O}\left(\frac{h}{m}^{k+1}\right)$$

$$\widehat{r}_{n+1} = \|\frac{y^h_{n+1} - y^{\frac{h}{m}}_{n+1}}{1 - m^{-k}}\|$$

Elementary Error Control

Controller, see SÖDERLIND

$$h_{n+1} = \left(\frac{\epsilon}{\widehat{r}_{n+1}} \right)^{\frac{1}{k}} h_n$$

Elementary Error Control

Controller, see SÖDERLIND

$$h_{n+1} = \left(\frac{\epsilon}{\widehat{r}_{n+1}} \right)^{\frac{1}{k}} h_n$$

Properties

- error estimate larger $\epsilon \Rightarrow$ decrease time step size
- error estimate smaller $\epsilon \Rightarrow$ increase time step size

Integral Controller

$$h_{n+1} = \left(\frac{\epsilon}{\widehat{r}_{n+1}} \right)^{k_I} h_n$$

Integral Controller

$$h_{n+1} = \left(\frac{\epsilon}{\widehat{r}_{n+1}} \right)^{k_I} h_n$$

taking logarithm on both sides

Integral Controller

$$h_{n+1} = \left(\frac{\epsilon}{\widehat{r}_{n+1}} \right)^{k_I} h_n$$

taking logarithm on both sides

$$\log\, h_{n+1} = \log\, h_n + k_I(\log\, \epsilon - \log\, \widehat{r}_{n+1})$$

Integral Controller

$$h_{n+1} = \left(\frac{\epsilon}{\widehat{r}_{n+1}} \right)^{k_I} h_n$$

taking logarithm on both sides

$$\log\, h_{n+1} = \log\, h_n + k_I(\log\, \epsilon - \log\, \widehat{r}_{n+1})$$

- $\log\, \epsilon - \log\, \widehat{r}_{n+1}$: *control error*
- ϵ *setpoint*
- k_I *integral gain*

Integral Controller

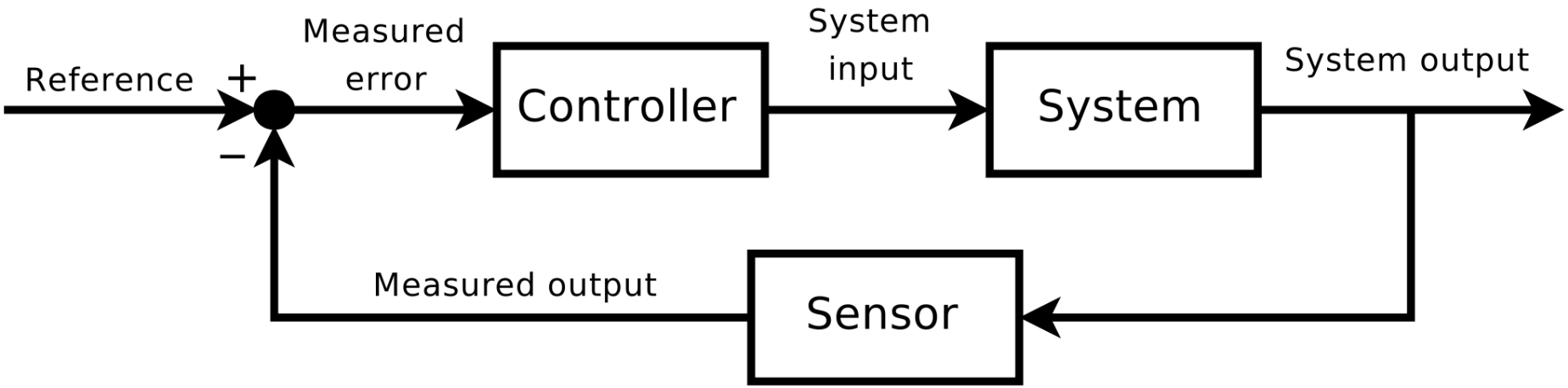
$$h_{n+1} = \left(\frac{\epsilon}{\widehat{r}_{n+1}} \right)^{k_I} h_n$$

taking logarithm on both sides

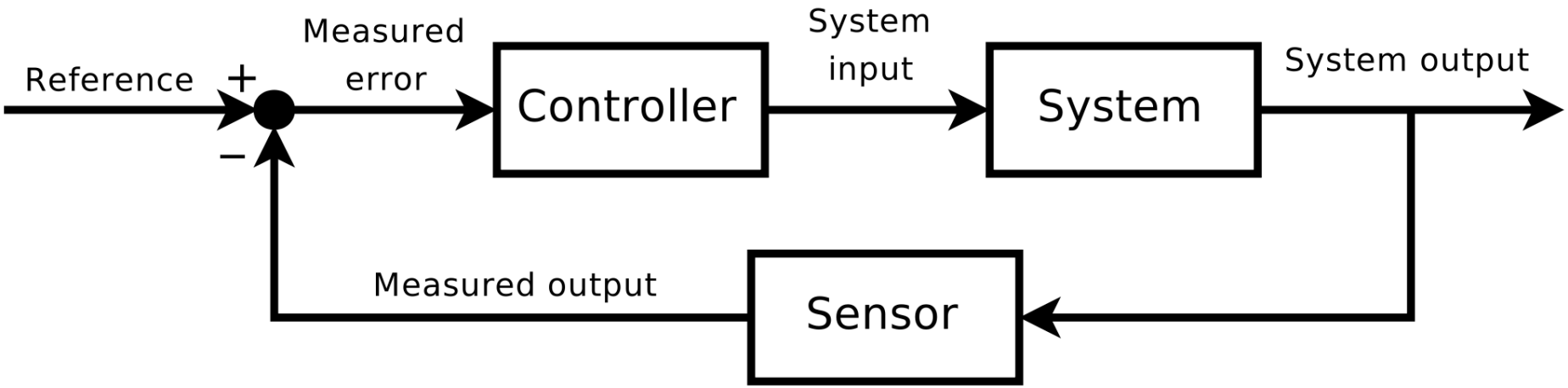
$$\log\, h_{n+1} = \log\, h_n + k_I(\log\, \epsilon - \log\, \widehat{r}_{n+1})$$

- $\log\, \epsilon - \log\, \widehat{r}_{n+1}$: *control error*
- ϵ *setpoint*
- k_I *integral gain* ($k_I k \in [0, 2]$ for stability)

Control Theory Model



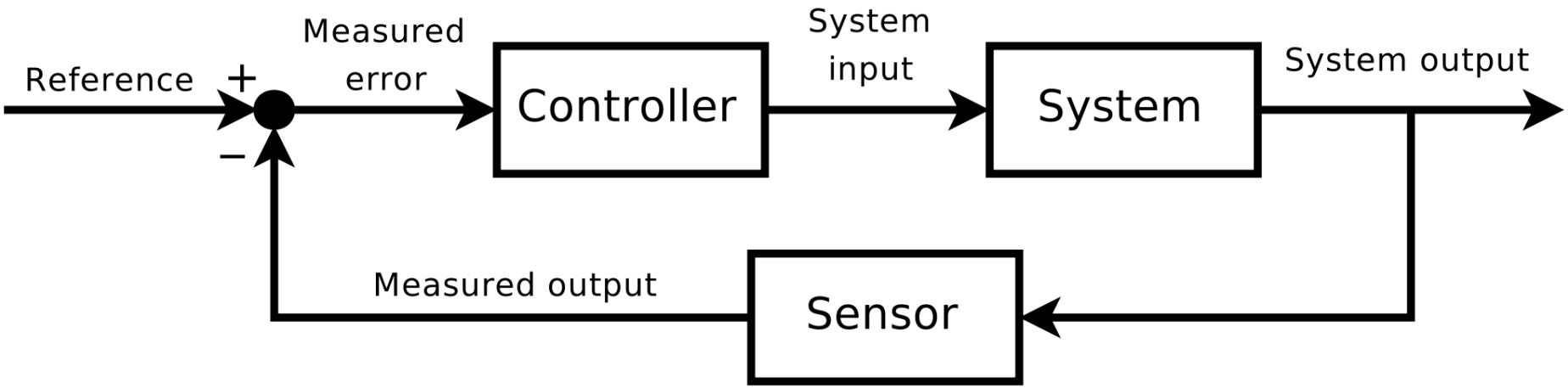
Control Theory Model



Questions

- when is the whole system stable?

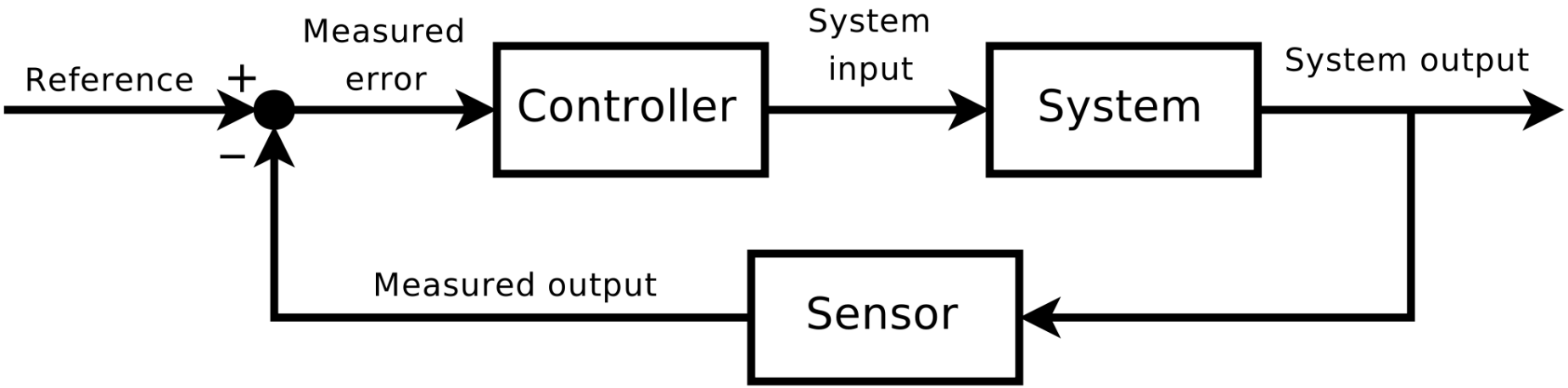
Control Theory Model



Questions

- when is the whole system stable?
- which controllers can we use?

Control Theory Model



Questions

- when is the whole system stable?
- which controllers can we use?

Other Controllers

Integral controller

$$h_{n+1} = \left(\frac{\epsilon}{\widehat{r}_{n+1}} \right)^{k_I} h_n$$

Other Controllers

Integral controller

$$h_{n+1} = \left(\frac{\epsilon}{\widehat{r}_{n+1}}\right)^{k_I} h_n$$

Proportional-integral controller

$$h_{n+1} = \left(\frac{\epsilon}{\widehat{r}_{n+1}}\right)^{k_I} \left(\frac{\widehat{r}_n}{\widehat{r}_{n+1}}\right)^{k_P} h_n$$

Other Controllers

Integral controller

$$h_{n+1} = \left(\frac{\epsilon}{\widehat{r}_{n+1}}\right)^{k_I} h_n$$

Proportional-integral controller

$$h_{n+1} = \left(\frac{\epsilon}{\widehat{r}_{n+1}}\right)^{k_I} \left(\frac{\widehat{r}_n}{\widehat{r}_{n+1}}\right)^{k_P} h_n$$

Predictive controller

$$\frac{h_{n+1}}{h_n} = \left(\frac{\epsilon}{\widehat{r}_{n+1}}\right)^{k_E} \left(\frac{\widehat{r}_n}{\widehat{r}_{n+1}}\right)^{k_R} \frac{h_n}{h_{n-1}}$$

Non-Linear System

$$\frac{\partial Q}{\partial t} + R(Q) = 0$$

Non-Linear System

$$\frac{\partial Q}{\partial t} + R(Q) = 0 \qquad \Rightarrow \qquad \widetilde{R}(Q^{n+1}) = 0$$

Non-Linear System

$$\frac{\partial Q}{\partial t} + R(Q) = 0 \qquad \Rightarrow \qquad \widetilde{R}(Q^{n+1}) = 0$$

Iterative solution is necessary. Define the update

$$\Delta Q_{k+1}^{n+1} = Q_{k+1}^{n+1} - Q_k^{n+1}$$

Non-Linear System

$$\frac{\partial Q}{\partial t} + R(Q) = 0 \qquad \Rightarrow \qquad \widetilde{R}(Q^{n+1}) = 0$$

Iterative solution is necessary. Define the update

$$\Delta Q_{k+1}^{n+1} = Q_{k+1}^{n+1} - Q_k^{n+1}$$

Newton's algorithm

$$\frac{\partial \widetilde{R}}{\partial Q} \Delta Q_{k+1}^{n+1} = -\widetilde{R}(Q_k^{n+1})$$

Non-Linear System

$$\frac{\partial Q}{\partial t} + R(Q) = 0 \qquad \Rightarrow \qquad \widetilde{R}(Q^{n+1}) = 0$$

Iterative solution is necessary. Define the update

$$\Delta Q_{k+1}^{n+1} = Q_{k+1}^{n+1} - Q_k^{n+1}$$

Newton's algorithm

$$\frac{\partial \widetilde{R}}{\partial Q} \Delta Q_{k+1}^{n+1} = -\widetilde{R}(Q_k^{n+1})$$

Dual time stepping

$$\left(\frac{1}{\Delta \tau} I + \frac{\partial \widetilde{R}}{\partial Q} \right) \Delta Q_{k+1}^{n+1} = -\widetilde{R}(Q_k^{n+1})$$

Computation of Jacobian

Non-linear solver needs Jacobian of discretized residual function $\widetilde{R}$

Computation of Jacobian

Non-linear solver needs Jacobian of discretized residual function $\widetilde{R}$

Analytical computation

+ exact derivative
− error prone, tedious

Computation of Jacobian

Non-linear solver needs Jacobian of discretized residual function $\widetilde{R}$

Analytical computation

+ exact derivative
– error prone, tedious

Finite differences

+ arbitrary right hand side function
– only approximation of derivative
– number of right hand side evaluations increases with unknowns

Computation of Jacobian

Non-linear solver needs Jacobian of discretized residual function $\widetilde{R}$

Analytical computation

+ exact derivative
– error prone, tedious

Finite differences

+ arbitrary right hand side function
– only approximation of derivative
– number of right hand side evaluations increases with unknowns

Solution

efficient finite differences

Solution of linear system

Solution of linear system

Iterative solver

- GMRES
- BiCG
- BiCGSTAB

Solution of linear system

Iterative solver

- GMRES
- BiCG
- BiCGSTAB

Preconditioner

- SSOR
- ILU

Results

Testcase

Testcase

Linear test

- linear heat equation
- square domain
- quasi 1D setting, Dirichlet BC $Q = 0$, IC $Q = 0$
- right hand side sine function

Testcase

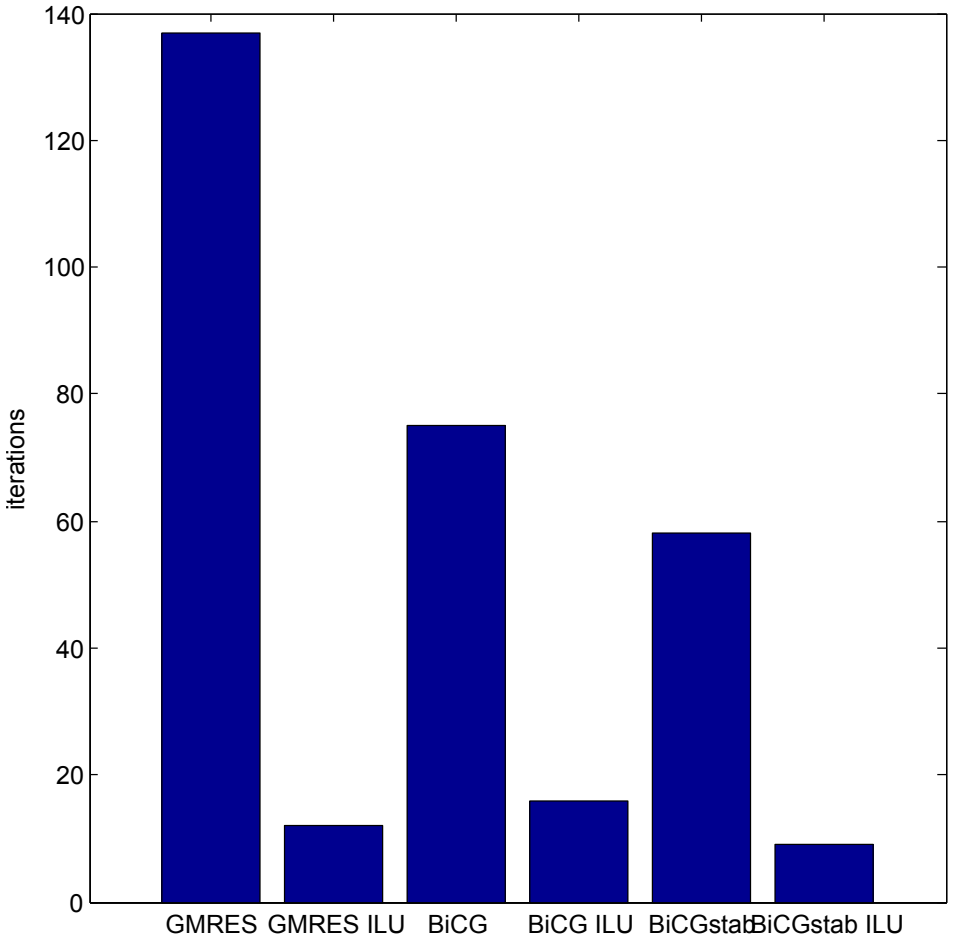
Linear test

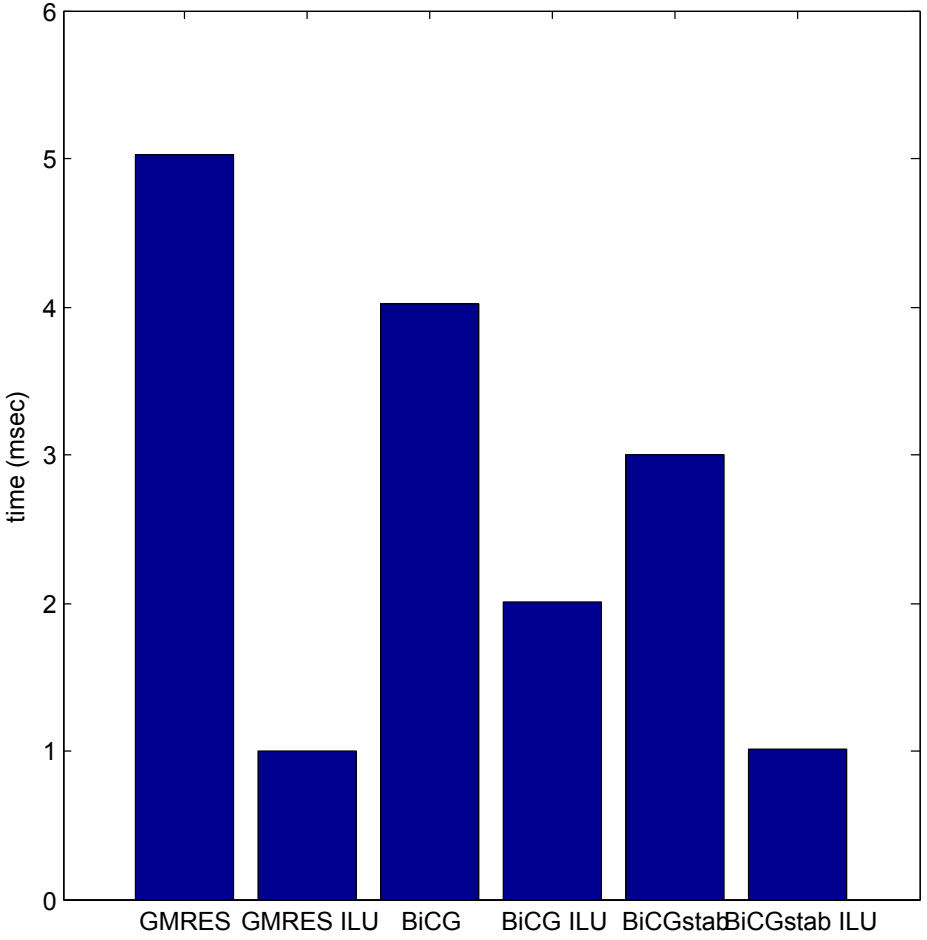
- linear heat equation
- square domain
- quasi 1D setting, Dirichlet BC $Q = 0$, IC $Q = 0$
- right hand side sine function

Non-linear test

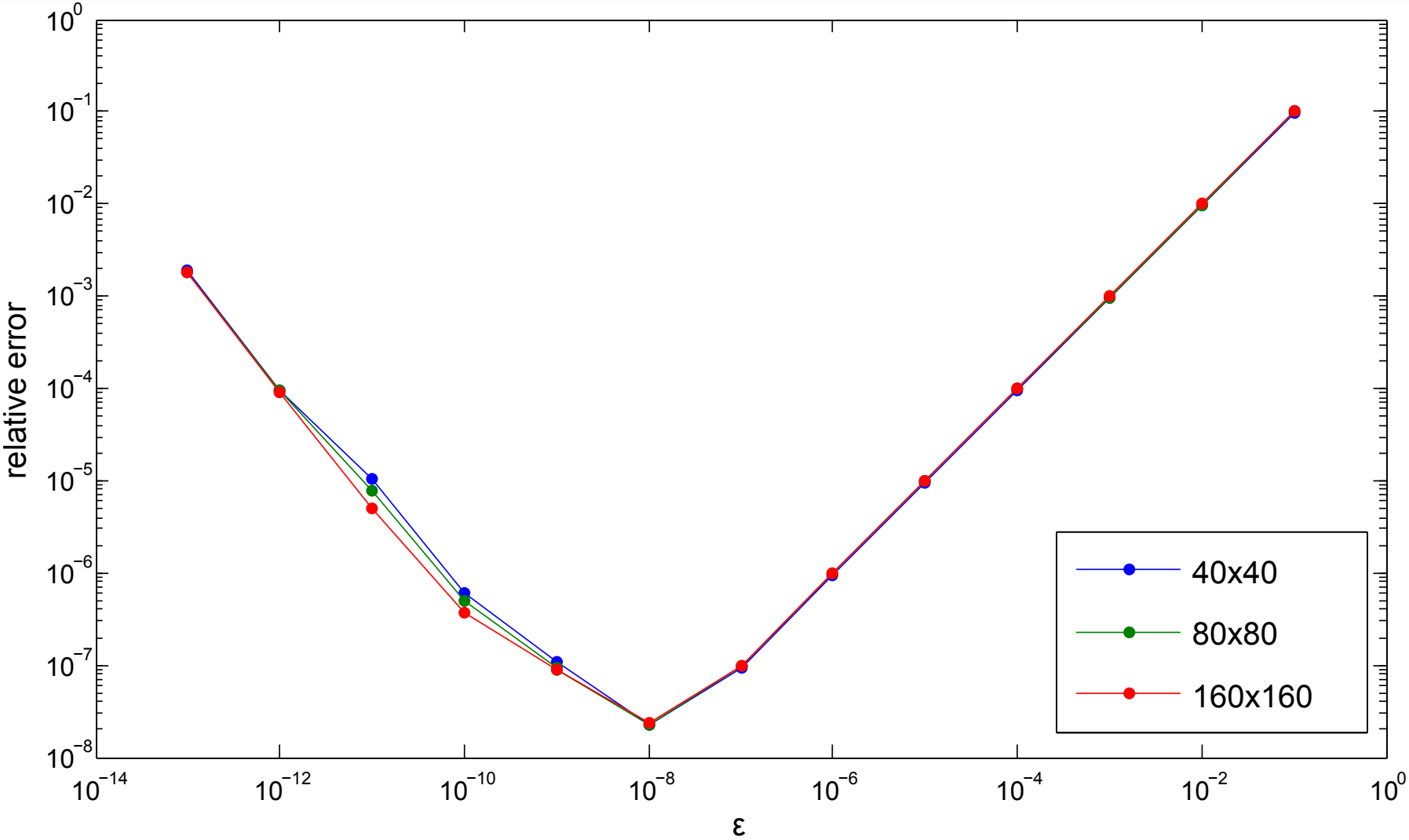
- non-linear heat equation
- square domain
- Dirichlet BC $Q = 0$, IC $Q = 1$
- with constant right hand side function

Linear solver iterations and runtime

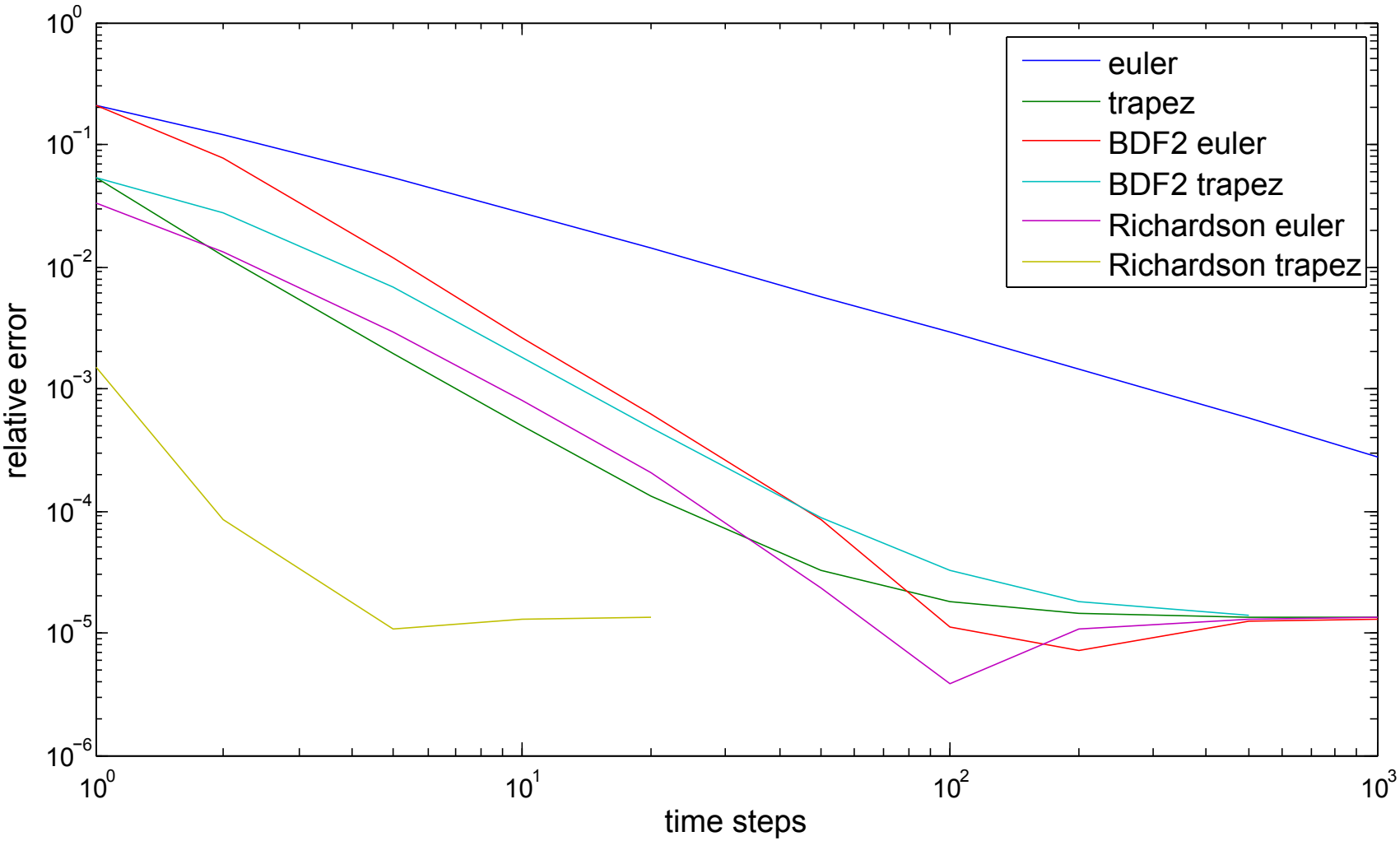




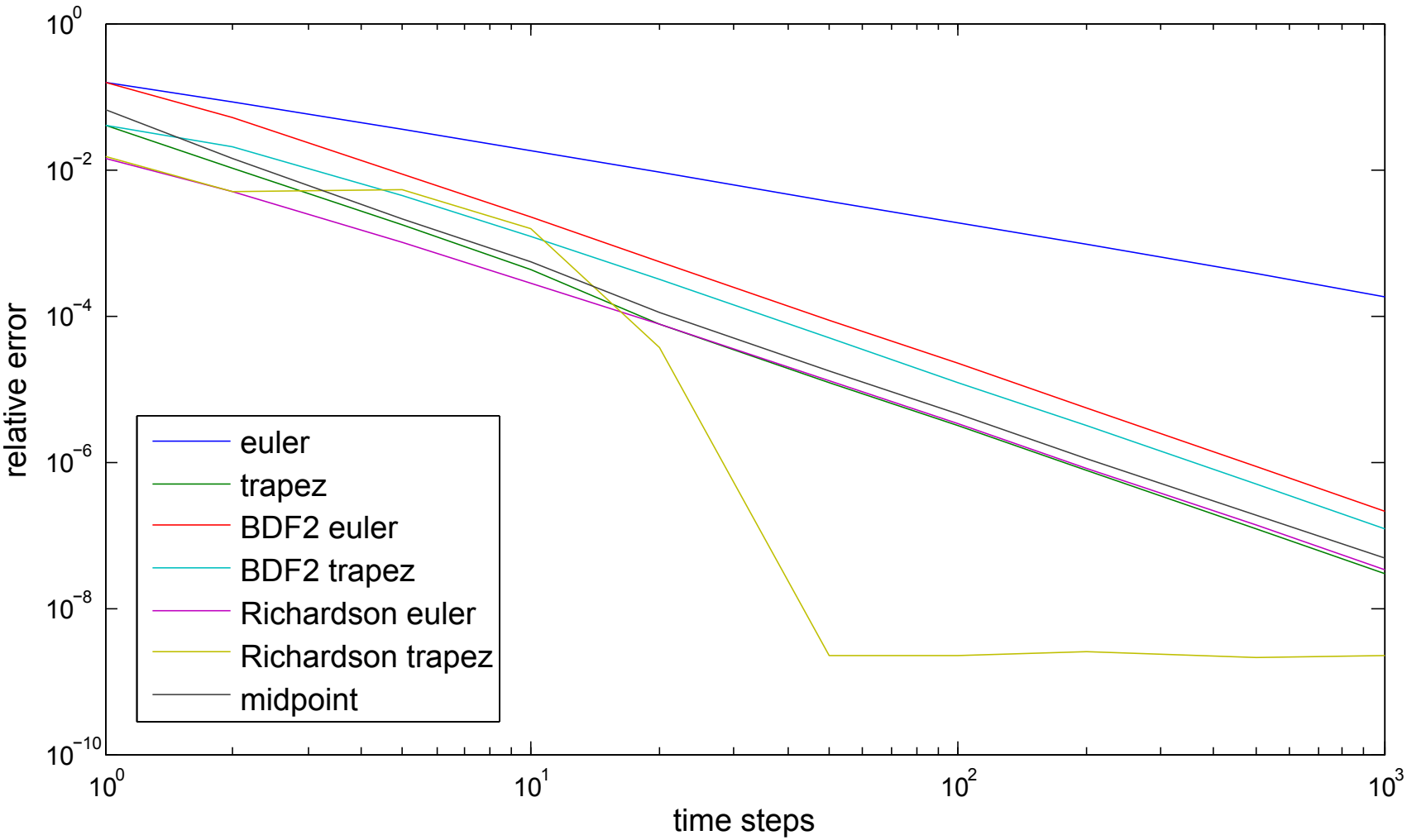
Error of Jacobian calculation



Linear test case



Non-linear test case



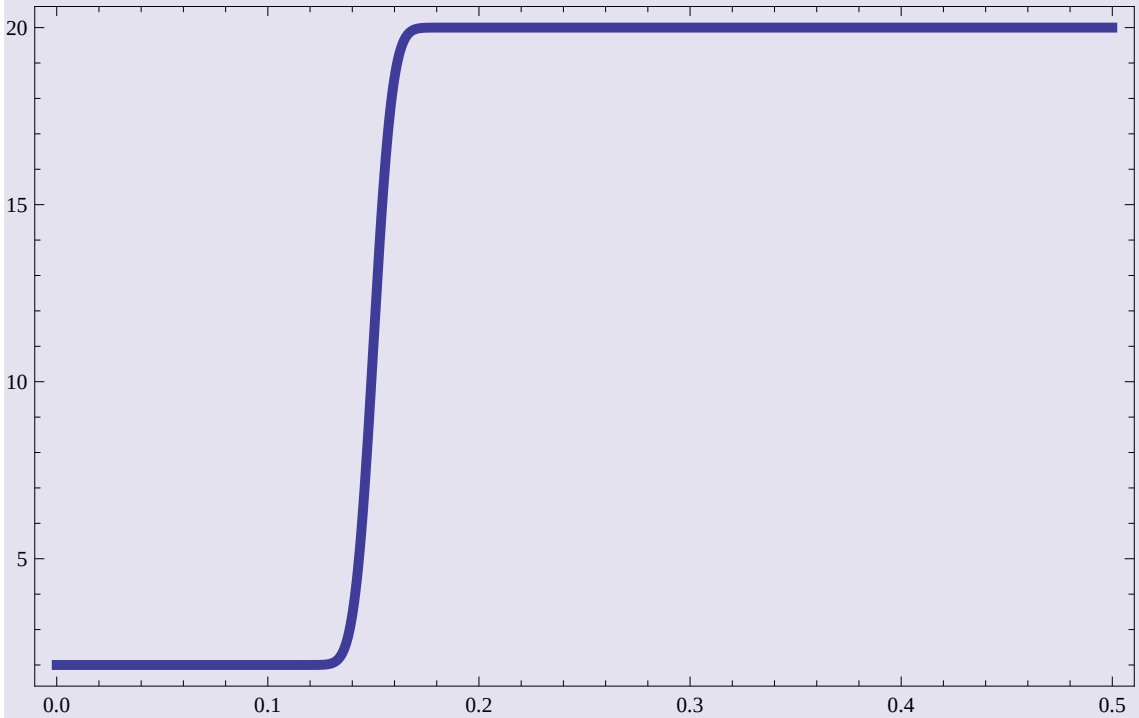
Test case

Linear heat equation with right hand side function

Test case

Linear heat equation with right hand side function

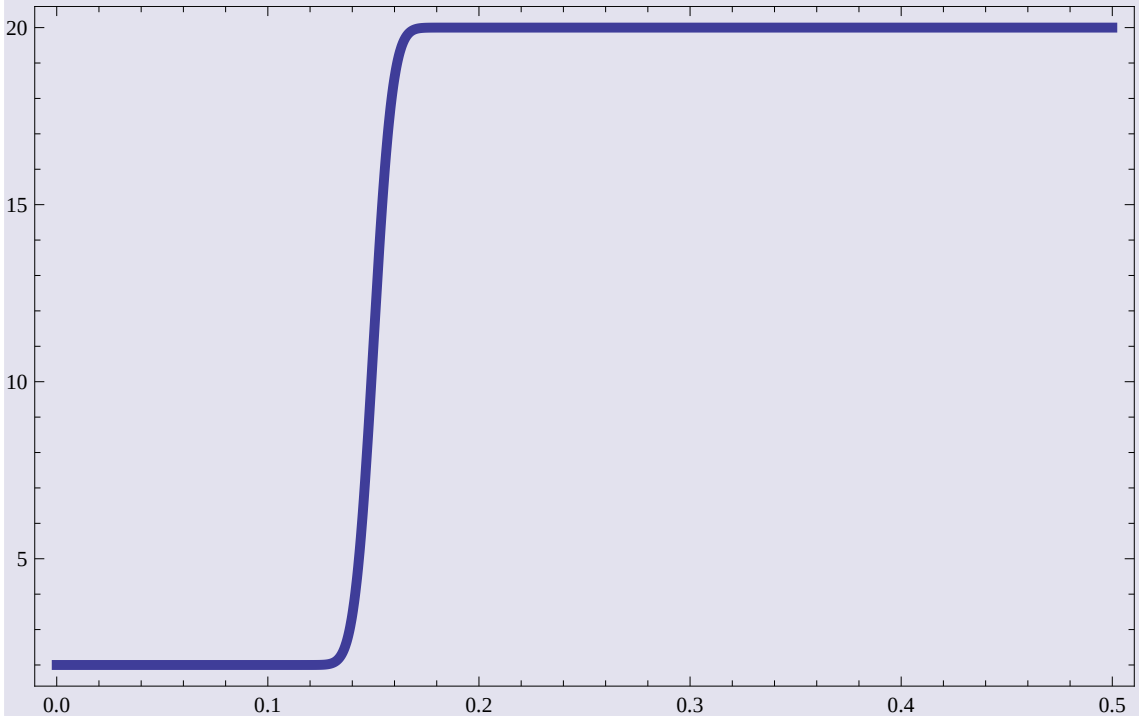
Non-linear right hand side function



Test case

Linear heat equation with right hand side function

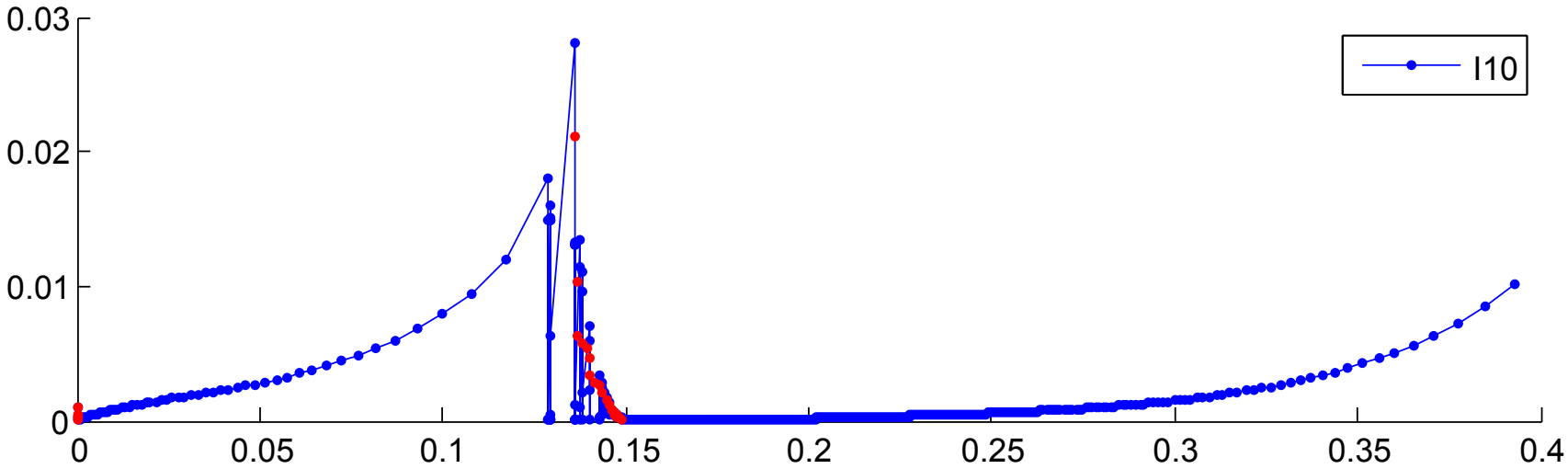
Non-linear right hand side function



Time step development

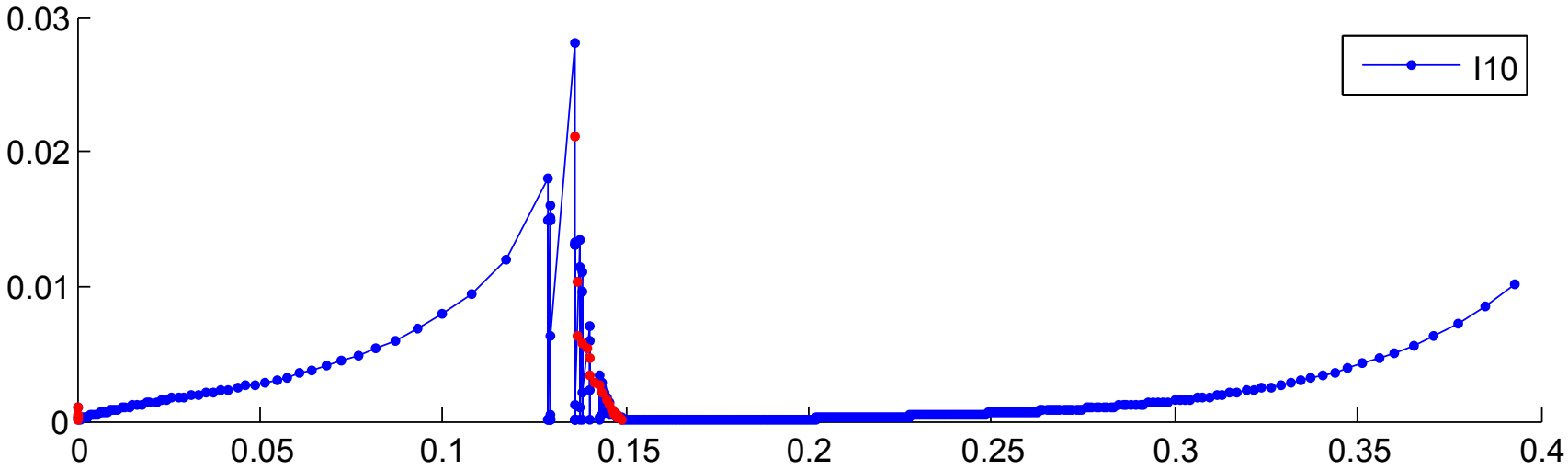
- increasing time step at the beginning
- strong non-linearity near jump at $Q = 0.15$ leads to small timesteps
- large time steps in the end

I vs. PI controller

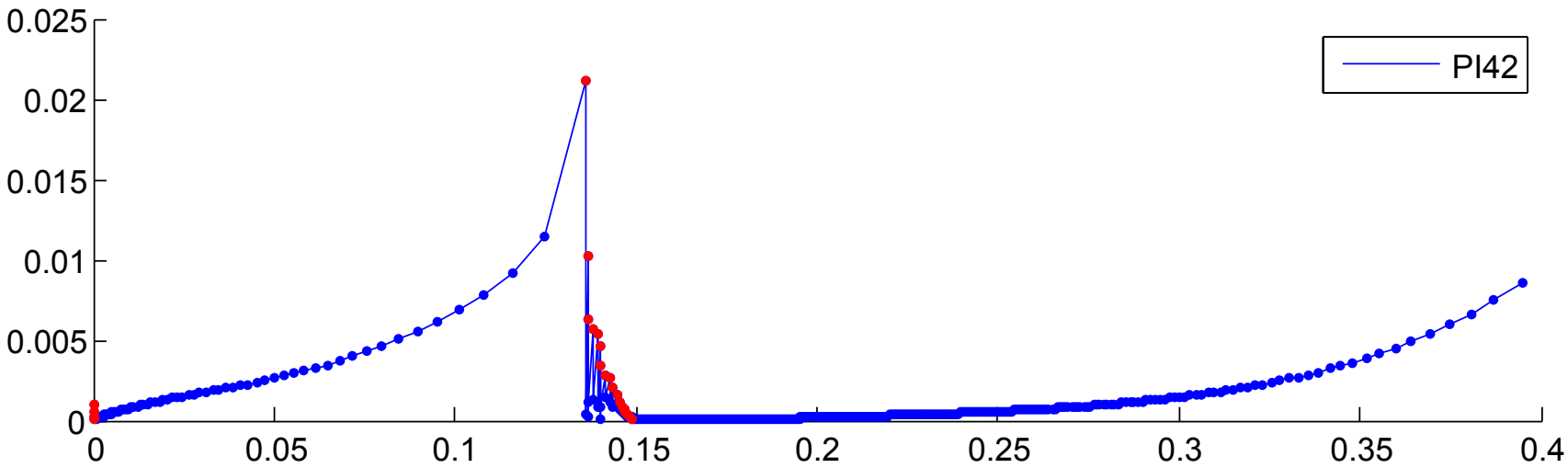


timesteps:
1170
rejects:
34

I vs. PI controller

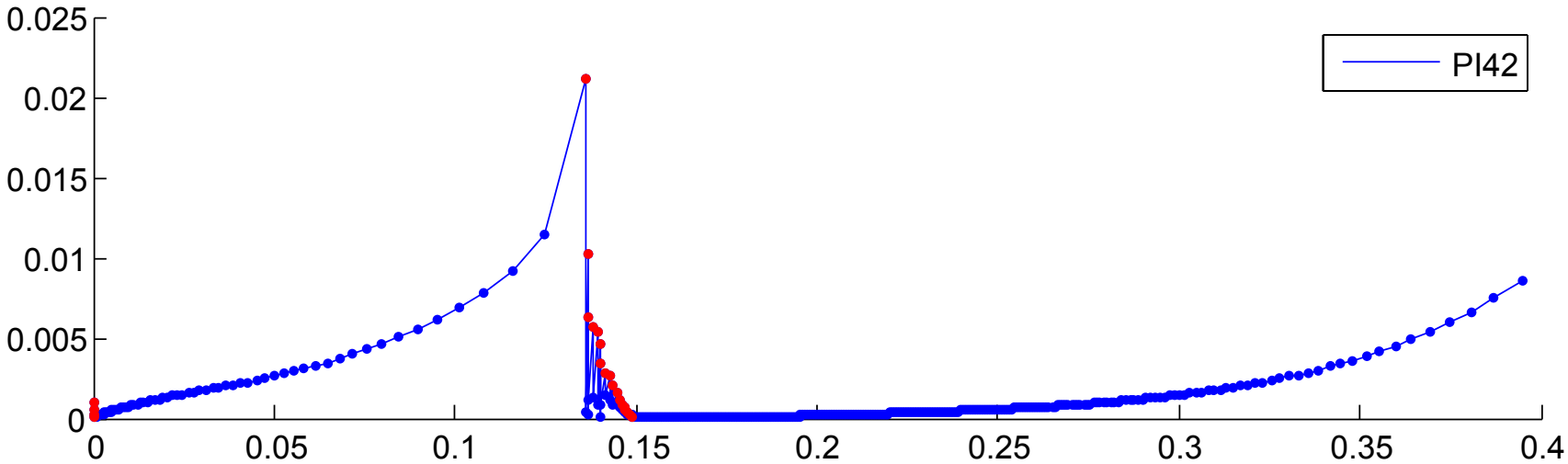


timesteps:
1170
rejects:
34



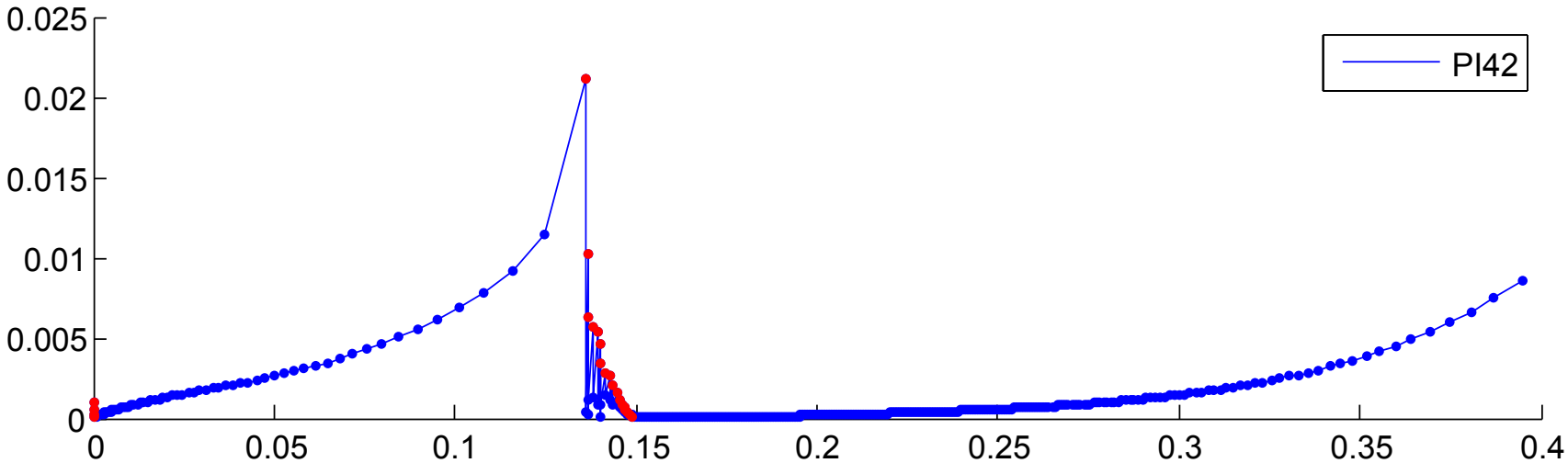
timesteps:
1169
rejects:
32

PI vs. PC controller

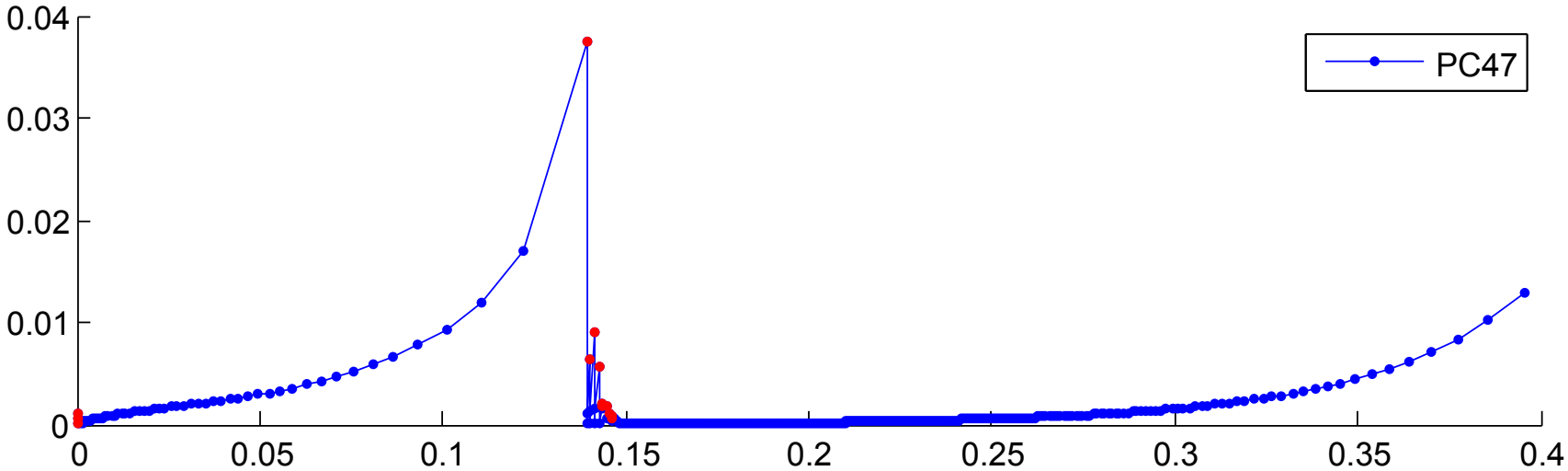


timesteps:
1169
rejects:
32

PI vs. PC controller



timesteps: 1169
rejects: 32

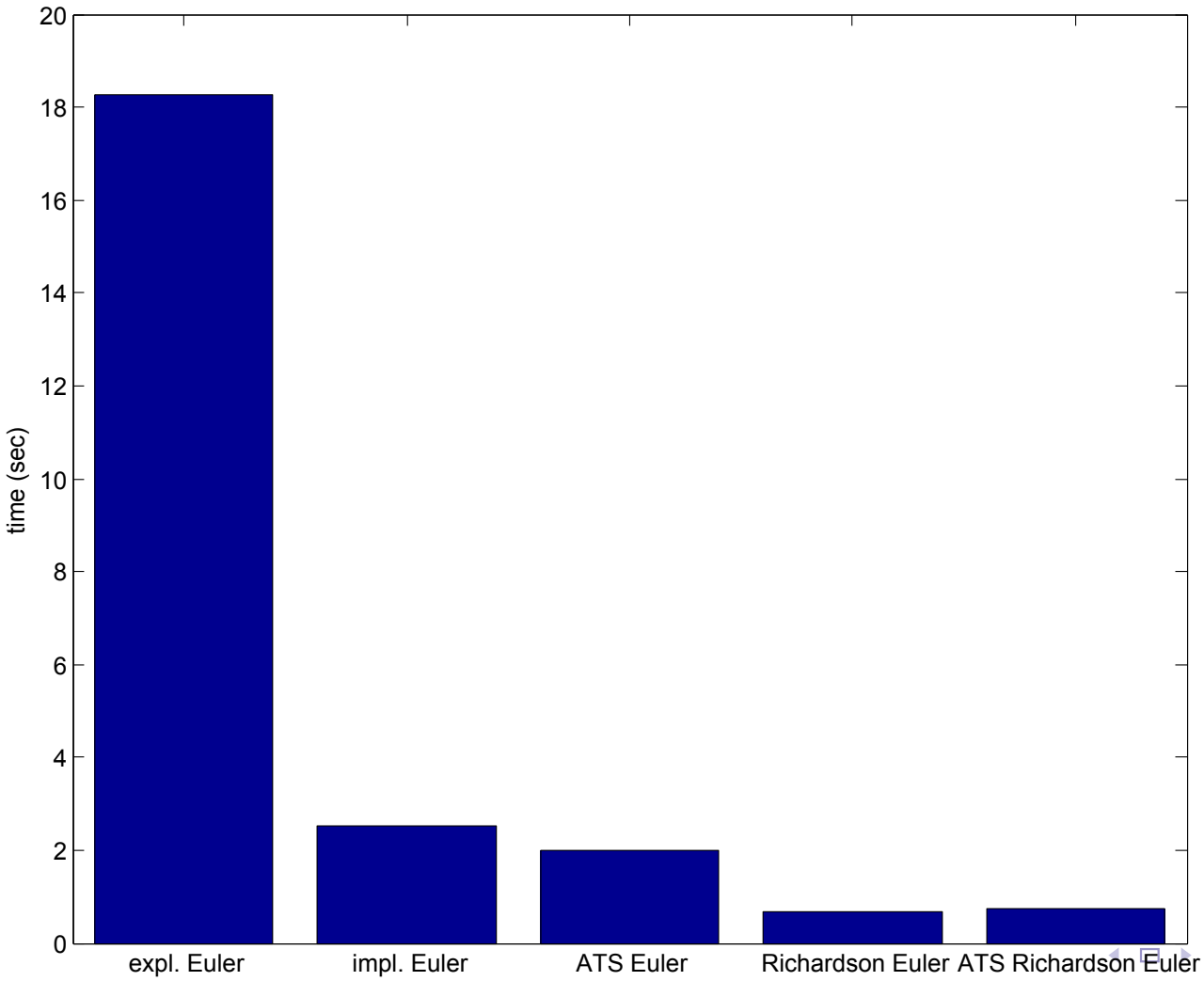


timesteps: 1128
rejects: 14

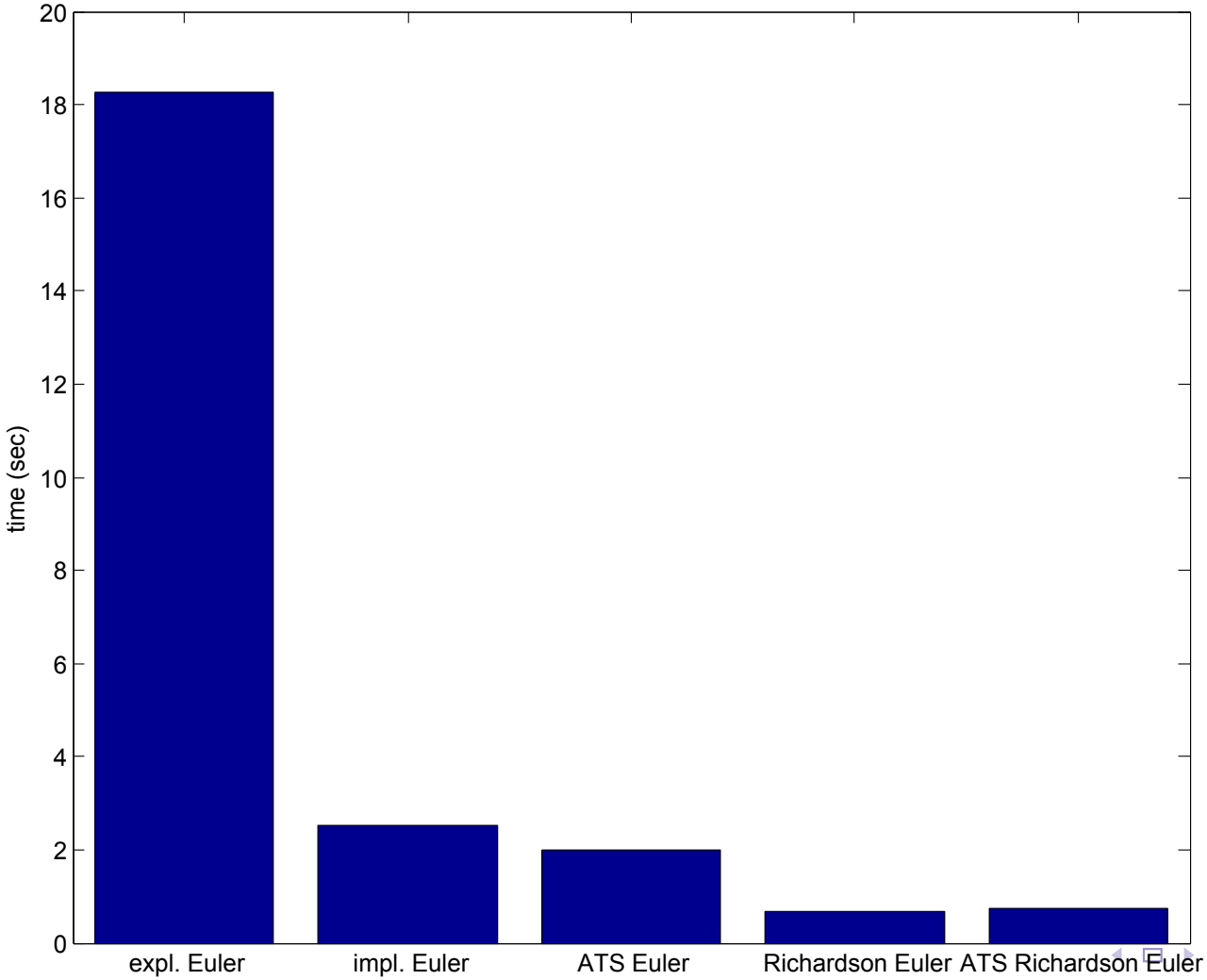
Test setting

- non-linear heat equation with constant right hand side
- calculate Q at time $t_{end} = 1sec$
- expl. method: time step size limited by stability
- impl. methods: maximum time step size for 1% relative error

Runtime Measurements



Runtime Measurements



expl. Euler
18.2 sec

impl. Euler
2.5 sec

ATS impl. Euler
2.0 sec

Richardson Extrapolation
0.7 sec

Conclusion

Conclusion

Successful implementation and comparison of different implicit time stepping techniques

- different implicit time discretizations
- adaptive time stepping
- non-linear and linear solvers as well as preconditioners
- comparison of methods for acceleration of simulation
- speedup of more than 90% with respect to explicit method

Thank you for your attention