NTNU
Department of Mathematical Sciences

To be filled in

Sindre Eskeland

December 11, 2015

MASTER

Department of Mathematical Sciences

Norwegian University of Science and Technology

Supervisor: Professor Elena Celledoni

Preface

To be filled in!

Acknowledgment

To be filled in.

Summary and Conclusions

To be filled in!

Contents

Chapter 1

Introduction

Skrive litt om problemet, layouten og slikt.

!!!!!!!!!!!!!!!!!!!!!!cite rapport og SLM !!!!!!!!!!!!!!!

!!Cite hamiltonian!!!!!!!!!!!!!!!!!!!

The equation

$$\begin{aligned} \dot{u}(t) &= Au(t) + F(t) = g(t) \\ u(0) &= u_0 \end{aligned} \tag{1.1}$$

often makes an appearance when solving partial differential equations with numerical methods. The author has earlier observed how the heat equation, discretized with finite difference methods to be on the form of equation (1.1) can be solved with the use of the Krylov projection method(KPM) [2]. This note will continue on the same track, with more focus on the wave equation, and energy preservation. It will also feature a comparison between symplectic Lanzcos method(SLM) [6] and KPM. SLM is a projection technique that only works on Hamiltonian matrices. Due to this, SLM (claim to) preserve energy better than KPM. This note will also compare time consumption and error for the different methods, but not much theoretical derivation. For this I recommend reading [1], [2], and [3].

!!!!!!!!!!!!!!!!!!MÅ cite forsjkellig når jeg vil/anbefaler å lese hele, og når jeg anbefaler å lese deler av det?!!!!!!!!!

Chapter 2

Explonation

!!!!!!!!!!!!!!!!Alt må leses igjennom!!!!!!!!!!!!!!!!

!!!!!!!!!!!!!!!!!!Citinger må fikses!!!!!!!!!!!!!!!!!!

!!!!!!!!!!!!!!!!Bilder må oppdateres!!!!!!!!!!!!!!!!!!!!!!

!!!!!!!!!!!!!!!!!!!!Det må skrives en del tekts mange steder!!!!!!!!!!!!!!!!

!!!!!!!!!!!!!!!!og bedre forklaringer mange steder!!!!!!!!!!!!!!!!!!!!!!

!!!!!!!!!!!!!!!!mye å ta tak i med andre ord!!!!!!!!!!!!!!!!

!!!!!!!!!!!!!!!!!!KPM er ikke det samme som SLM?!!!!!!!!!!!!!!!!!!

!!!!!!!!!!!!!!!!!!Størrelse på J?!!!!!!!!!!!!!!!!!!!!!!!!! There will here be a short explanation of all solvers, constants, out data and expressions used in this text. MATLAB notation is used where applicable.

2.1 Projection methods

As mentioned earlier, two projection methods will be presented, KPM and SLM. They are very similar in nature, with the only difference being the orthogonalisation method. KPM uses Arnoldi's algorithm, given in Algorithm 1, and SLM uses the symplectic Lanczos method, given in Algorithm 2. The framework for both methods is given in Algorithm 3.

Note that the relation between $\hat{m}$, $\tilde{n}$, $\tilde{m}$ used it the algorithms is given by $\hat{m} = 2\tilde{m} = 2(m-2)^2$ and $n = 2\tilde{n}$.

The approximated solution, found by either SLM or KPM, will be denoted by u^n, where n the

Algorithm 1 Arnoldi's algorithm[10]

Start with $A \in \mathbb{R}^{\hat{m}\times\hat{m}}$, $v \in \mathbb{R}^{\hat{m}}$, $n \in \mathbb{N}$ and $\epsilon \in \mathbb{R}$.
$v_1 = v/\|v\|_2$
for $j = 1,2,\cdots,n$ **do**
  Compute $h_{i,j} = v_i A v_j, v_i$ for $i = 1,2,\cdots,j$
  Compute $w_j = Av_j - \Sigma_{i=1}^{j} h_{i,j} v_i$
  $h_{j+1,j} = \|w_j\|_2$
  if $h_{j+1,j} < \epsilon$ **then**
    STOP
  end if
  $v_{j+1} = w_j / h_{j+1,j}$
end for
Return H, V, v_{n+1}, $h_{n+1,n}$.

Algorithm 2 Symplectic Lanczos method [6], with reortogonalization from [5].

Start with a Hamiltonian matrix $A \in \mathbb{R}^{2\tilde{m}\times 2\tilde{m}}$, $\tilde{v}_1 \in \mathbb{R}^{2\tilde{m}}$, $\tilde{n} \in \mathbb{N}$
$v_0 = 0 \in \mathbb{R}^{2\tilde{m}}$
$\xi_1 = \|\tilde{v}_1\|_2$
$v_1 = \frac{1}{\xi_1}\tilde{v}_1$
for $j = 1,2,\cdots,\tilde{n}$ **do**
  $v = Av_j$
  $\delta_j = v_j^\top$
  $\tilde{w} = v - \delta_j v_j$
  $\kappa_j = v_j^\top J_{\tilde{m}} v$
  $w_j = \frac{1}{\kappa_j}\tilde{w}_j$
  $w = Aw^j$
  $\tilde{V}_{j-1} = [v_1, v_2, \cdots, v_{j-1}, w_1, w_2, \cdots, w_{j-1}]$
  $w_j = w_j + \tilde{V}_{j-1} J_{j-1} \tilde{V}_{j-1}^\top J_{\tilde{m}} w_j$
  $\beta = -w_j^\top J_{\tilde{m}} w$
  $\tilde{v}_{j+1} = w - \xi_j v_{j-1} - \beta_j v_j + \delta_j v_j$
  $\xi_{j+1} = \|\tilde{v}_{j+1}\|_2$
  $v_{j+1} = \frac{1}{\xi_{j+1}}\tilde{v}_{j+1}$
  $\tilde{V}_j = [v_1, v_2, \cdots, v_j, w_1, w_2, \cdots, w_j]$
  $v_{j+1} = v_{j+1} + \tilde{V}_j J_j \tilde{V}_j^\top J_{\tilde{m}} v_{j+1}$
end for
$V = [v_1, v_2, \cdots, v_{\tilde{n}}, w_1, w_2, \cdots, w_{\tilde{n}}]$
$$H = \begin{bmatrix} \mathrm{diag}\left([\delta_j]_{j=1}^{n}\right) & \mathrm{tridiag}\left([\xi_j]_{j=2}^{n}, [\beta_j]_{j=1}^{n}, [\xi_j]_{j=2}^{n}\right) \\ \mathrm{diag}\left([\kappa_j]_{j=1}^{n}\right) & \mathrm{diag}\left([-\delta_j]_{j=1}^{n}\right) \end{bmatrix}$$
Return H, V, v_{n+1}, ξ_{n+1}.

Algorithm 3 Framework for the orthogonalisation methods[2]

Start with $A \in \mathbb{R}^{\hat{m}\times\hat{m}}$, $f(t)$, $v \in \mathbb{R}^{\hat{m}}$, $n \in \mathbb{N}$, a boolean value `restart`, an algorithm `alg`, $\epsilon \in \mathbb{R}$, and $i = 0$.
Compute $[V_n, H_n, h^i_{n+1,n}, v_{n+1}] = \texttt{alg}(A, v, n)$
Solve $z_i'(t) = H_n z_i(t) + f(t)\|v\|_2 e_1$ for $z_i(t)$
$u_n(t) \leftarrow V_n z_i(t)$
$\delta = h_{n+1,n}$
if `restart` $== 1$ **then**
  while $\epsilon < \delta$ **do**
    $i \leftarrow i + 1$
    Compute $[V_n, H_n, h^i_{n+1,n}, v_{n+1}] = \texttt{alg}(A, v_{n+1}, n)$
    Solve $z_i'(t) = H_n z_i(t) + h^{i-1}_{n+1,n} e_n^\top z_{i-1}(t)$ for $z_i(t)$
    $u_n(t) \leftarrow u_n(t) + V_n z_i(t)$
    $\delta = \max(u_n(t) - V_n z_i(t))$
  end while
end if
Return u_n.

same as the size for of the orthogonal space used (n).

If the solution is obtained without the use of an orthogonalisation, it will be denoted with DM for direct method.

2.2 Zero initial condition

For both KPM and SLM it is important that the initial conditions are zero. Equation (1.1) can be transformed so that it has zero initial conditions in the following way:

$$\hat{u}(t) = u(t) - u_0$$

The original equation can then be written as

$$\begin{aligned} \dot{\hat{u}}(t) &= A\hat{u}(t) + Au_0 + F(t) \\ \hat{u}(0) &= 0 \\ u(t) &= \hat{u} + u_0. \end{aligned} \tag{2.1}$$

All test problems with a non-zero initial condition will be transformed in this way.

2.3 Discretization

The number of points in each spacial direction is m, making the step size $h_s = 1/m$. The number of steps in time is k, making the step size $h_t = 1/k$.

Let the matrix I_j be the identity matrix of size j, and let

$$J_j = \begin{bmatrix} 0 & I_j \\ -I_j & 0 \end{bmatrix}. \tag{2.2}$$

Equation (1.1) can be the result of the discretization of several equation. Since SLM needs a Hamiltonian matrix this will be the main focus. One of the two implemented Hamiltonian matrices is the discretization of the 2 dimensional wave equation,

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + f(t,x,y). \tag{2.3}$$

This can be discretized to be on the form of equation (1.1), with the matrix

$$\begin{aligned} \tilde{A} &= \frac{2}{h_s^2}\,\text{gallery}('\text{poisson}', m-2) \\ A &= \begin{bmatrix} 0 & I_{\hat{m}} \\ -\tilde{A} & 0 \end{bmatrix} \end{aligned} \quad . \tag{2.4}$$

The matrix $\tilde{A}$ is also known as the five-point stencil[4]. This matrix will be referred to as `wave` when used. The second implemented Hamiltonian matrix is random, and given by

$$\begin{aligned} \hat{A} &= \text{rand}(\hat{m}) \\ A &= \frac{1}{2} J_{\hat{m}} (\hat{A} + \hat{A}^\top + m^2 I_{\hat{m}}). \end{aligned} \tag{2.5}$$

Since we are interested in comparing the different projection methods to each other, the matrix will be saved and reused when necessary. This matrix will be referred to as `semirandom`. The

part $m^2 I_{\hat{m}}$ is added to make $J_{\hat{m}} A$ diagonally dominant, there would be no way of knowing if any of the methods would converge without this part.

These two matrices also has some test problems that satisfies the condition: $u(t,0,y) = u(t,1,y) = u(t,x,0) = u(t,x,1) = 0$.

In the case when the energy is constant and `wave` is used, the test problem is

$$\begin{aligned} u(t,x,y) &= \sin(\pi x)\sin(\pi y)\cos(\sqrt{2}\pi t) \\ u_0(x,y) &= \sin(\pi x)\sin(\pi y) \\ f(t,x,y) &= 0, \end{aligned} \tag{2.6}$$

and

$$\begin{aligned} u(t,x,y) &= \text{unknown} \\ u_0(x,y) &= \text{rand}(2(m-2)^2, 1) \\ f(t,x,y) &= 0 \end{aligned} \tag{2.7}$$

for `semirandom`. This test problem is kept with the same conditions as A.

The source term $f(t,x,y)$ is chosen to be zero because it is easier to work with a constant energy. Some problems with non-constant energy will be presented later.

The test problems are discretized with $y_i = ih_s$, $x_i = ih_s$ and $t_j = jh_t$ with $i = 1,2,\cdots,m-1$ and $j = 1,2,\cdots,k$. The time discretized solution of u will be called U.

2.4 Lingo

Table 2.1 contains a short explanation of the labels you will see on figures later.

iterations	number of restarts performed by Arnoldi or symplectic Lanczos method.
time	computation time elapsed when solving the problem
error1	Difference in error between analytical solution, and estimated solution.
energy1	Difference between the largest and smallest energy during the simulated time.
m	number of point in each spacial direction
n	size of orthogonal space, also called restart variable
k	number of points in time
t	simulated time
`restart`	a boolean value. If `restart` == 1, Arnoldi or symplectic Lanczos method will restart.
ϵ	if `restart` is true, restarting will commence until the change in the solution is less than ϵ

Table 2.1: Explanation of some labels.

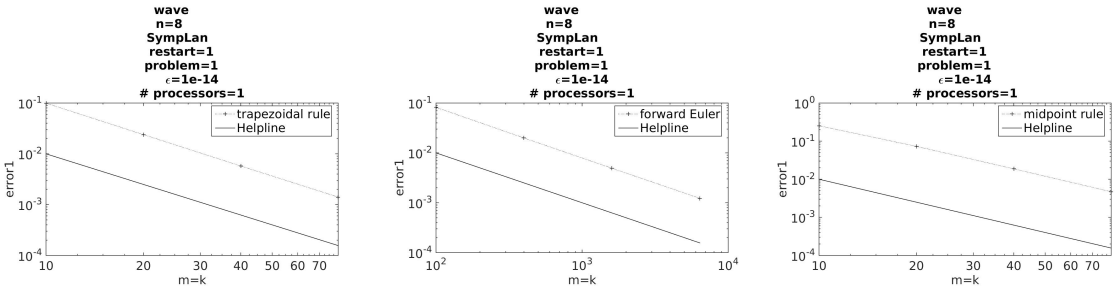
Chapter 3

Compare time integration methods

This section will be concerned with comparing the integrators. How well they estimate the error and energy will be the primary concern.

The relevant integrators are trapezoidal rule, forward euler, and midpoint rule. The definition, and the iteration scheme, of the different methods are given in the table below.

Convergence of the different methods are shown in figure 3.1

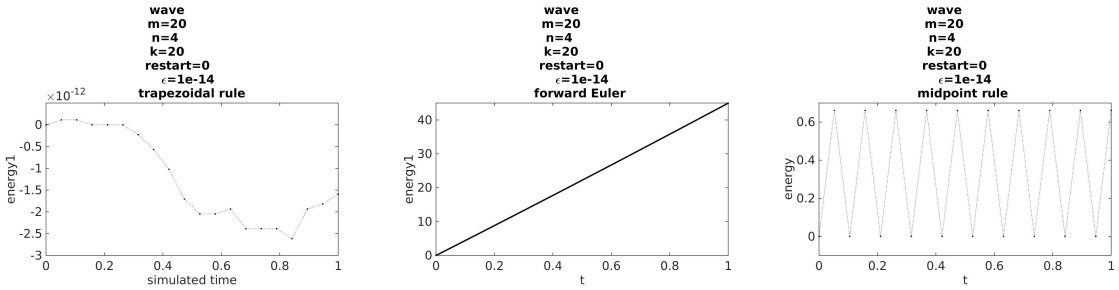


(a) Help line decreases with m^2. (b) Help line decreases with m. (c) Help line decreases with m^2.

Figure 3.1: Figure of the convergence for the different integration methods. All methods converge with the expected rate.

Trapezoidal rule [7]	$U_{i+1} = U_i + h_t g\left(\frac{1}{2}(t_i + t_{i+1}), \frac{1}{2}(U_i + U_{i+1})\right)$	$U_{i+1} = (I - \frac{Ah_t}{2}) \backslash \left(U_i + \frac{h_t}{2}\left(AU_i + (F_{i+1} + F_i)\right)\right)$
Forward Euler [8]	$U_{i+1} = U_i + h_t g(t_i, U_i)$	$U_{i+1} = U_i + h_t\left(AU_i + F_i\right)$
Midpoint rule [9]	$U_{i+2} = y_i + h_t + g\left(t_{i+1}, \frac{1}{2}(U_i + U_{i+2})\right)$	$U_{i+2} = (I - Ah_t) \backslash \left(U_i + 2h_t\left(\frac{AU_i}{2} + F_{i+1}\right)\right)$

Table 3.1: Proposed methods for integrating in time. A problem that arises with the midpoint rule is the need to know $F_{i+\frac{1}{2}}$, this is solved by doubling the step size. Er denne tabellen fin nok?



(a) (b) (c)

Figure 3.2: Figure of the difference in energy at a point in time. It is clear that only trapezoidal rule gives a suitable approximation of the error. Forward euler has an linearly increasing energy, while the midpoint rule gives periodic energy.

After looking at figure 3.2 it is easy to conclude that trapezoidal rule outperforms the other two integration methods.

Chapter 4

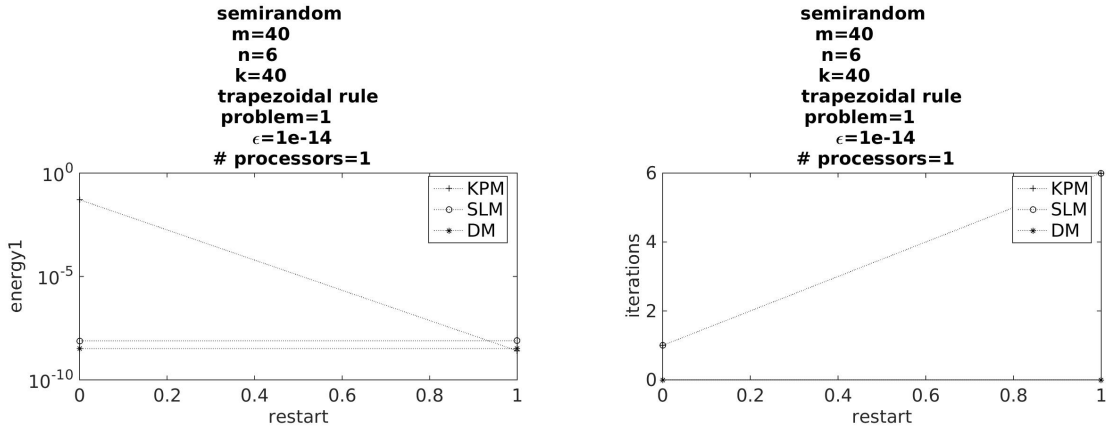
Energy preservation for SLM

It can be proved that a restart of SLM does not alter the energy [3]. This chapter is devoted to showing that this still hold with numerical approximations.

The residual energy of the symplectic Lanczos method is

$$\frac{1}{2}e_r^\top JAe_r + e_r^\top Jv_{n+1}e_{2\tilde{n}}^\top z \tag{4.1}$$

with $e_r = u - u_n$ (analytical solution minus approximated solution) and $v_{\tilde{n}+1}$ is the residual vector given by the symplectic Lanczos method.

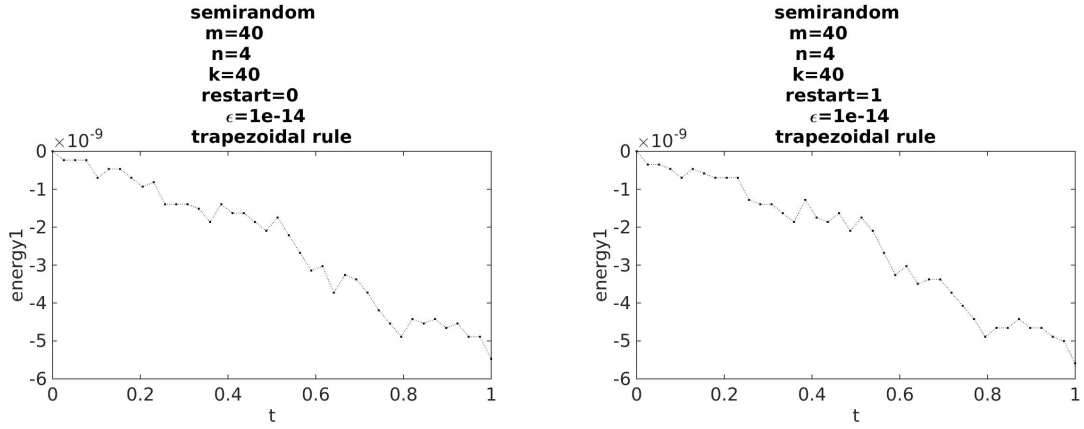


(a) The difference in energy with and without restart.

(b) The number of iterations performed with and without restarting.

Figure 4.1: The figure shows how the different methods change the energy with and without restarting.

The figures above implies that restarting the symplectic Lanczos method does indeed not change the energy. But for Arnoldi's method it changes quite a bit.

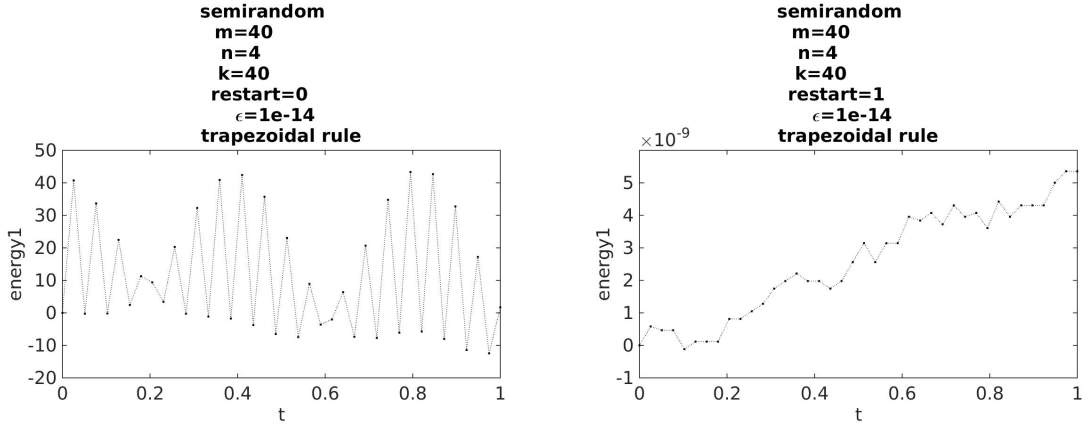


(a) Without restart. (b) With restart, the number of iterations = 47

Figure 4.2: The figures shows the change in energy over time.

The figure above shows very little change in energy with SLM, with and without several restarts.

The figures below shows the same as figure 4.2, but with KPM instead of SLM.



(a) Without restart. (b) With restart, the number of iterations = 48

Figure 4.3: Figures showing the change in energy over time.

When comparing figure 4.2a and 4.2b it becomes clear that SLM's does not alter the energy significantly when performing the restart. Figure 4.3a and 4.3b shows that this is not the case

for KPM.

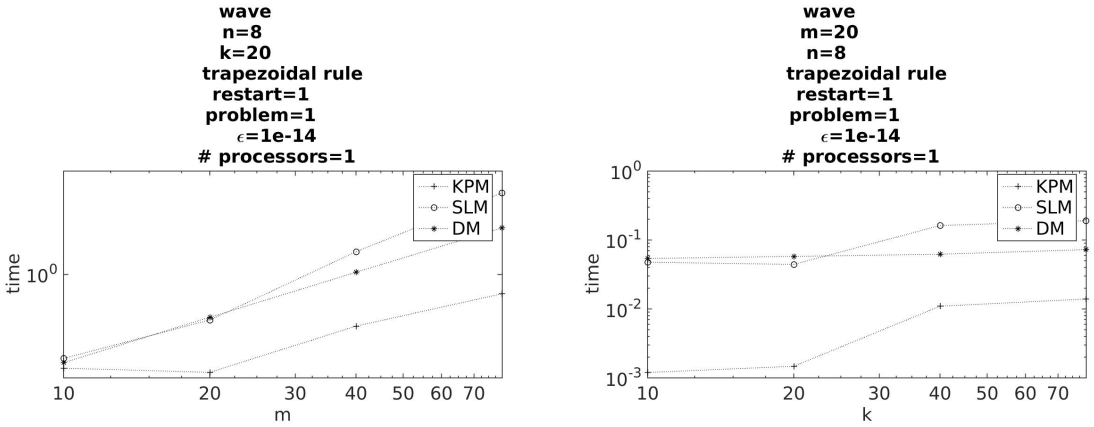
Chapter 5

Some interesting results

The energy has already been considered quite thoroughly, but there are still interesting factors to consider. This chapter will show how the different methods compare in relation to computation time, number of iterations, and error. This will be done with `wave` and `semirandom` both with constant and varying energy

5.1 Constant energy with the wave equation

This section will show how the the methods differs with constant energy and the wave equation.



(a) (b)

Figure 5.1: For both cases KPM is faster than the other, and with DM faster than SLM.

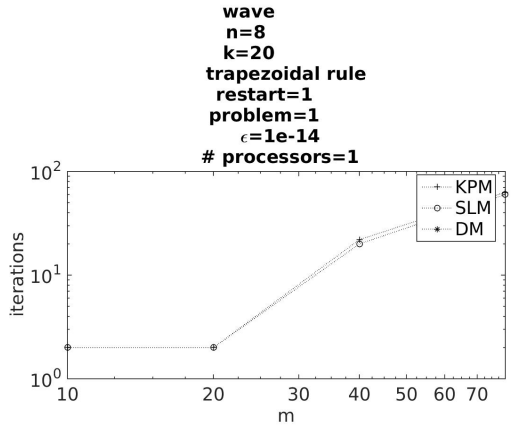
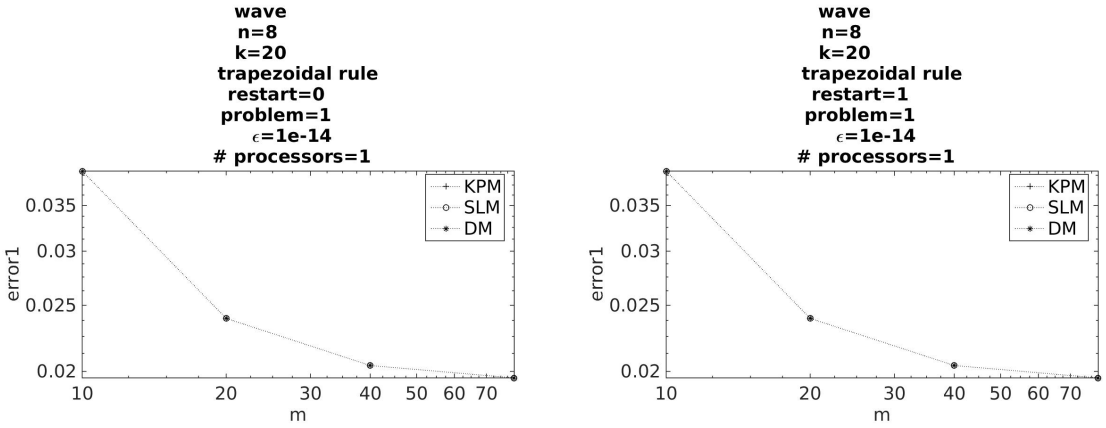
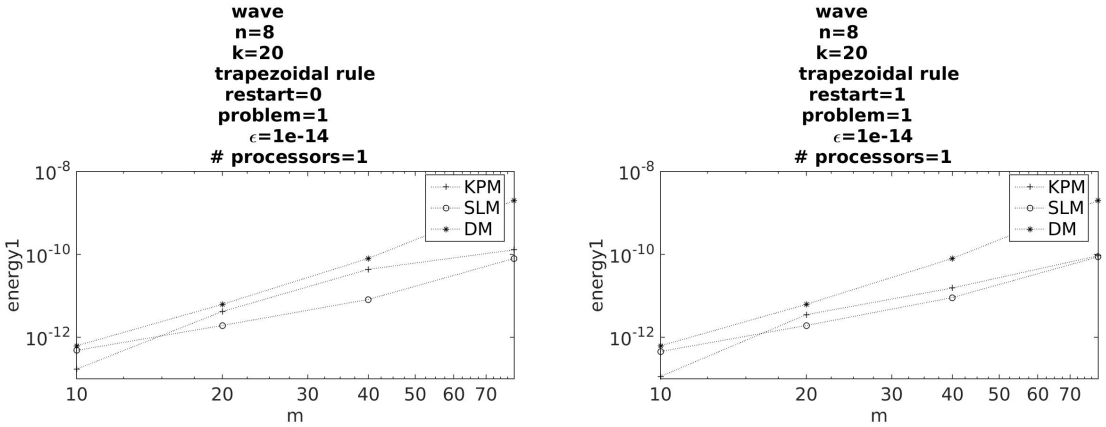


Figure 5.2: The number of iterations are almost equal for the two methods.



(a) (b)

Figure 5.3: The error is nearly identical.

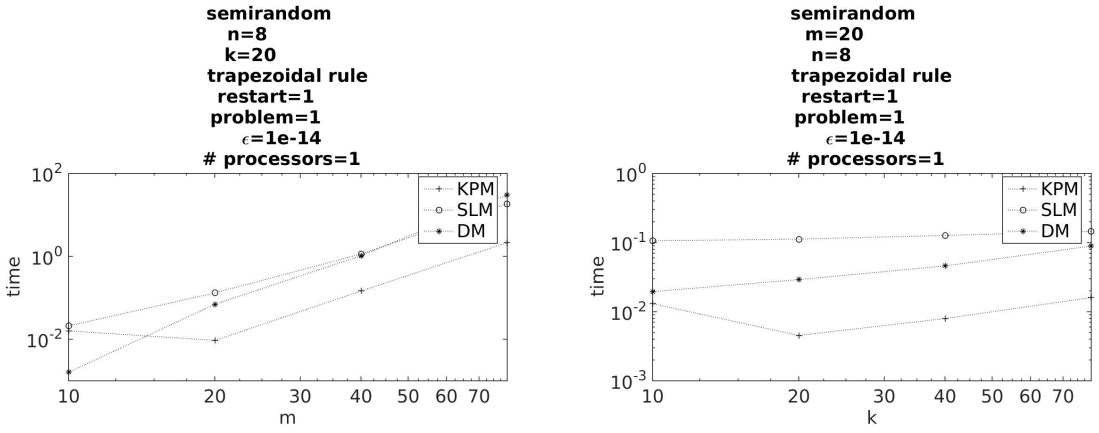


(a) (b)

Figure 5.4: SLM and KPM has the best energy estimation, with DM not far behind.

It is pretty clear that the figure in this section gives no useful information about the difference between the two methods in other tings than computation time.

5.2 Constant energy with the semirandom equation



(a) (b)

Figure 5.5: For both cases KPM is faster than the other, and with DM faster than SLM.

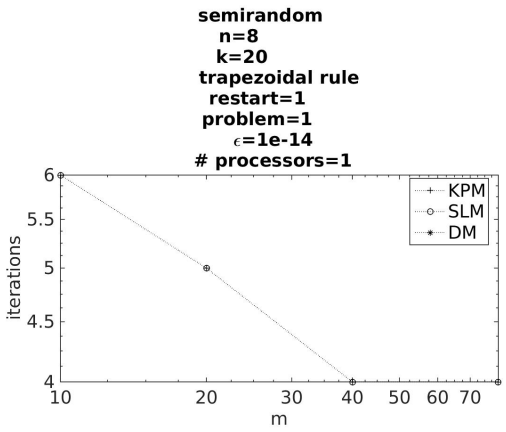
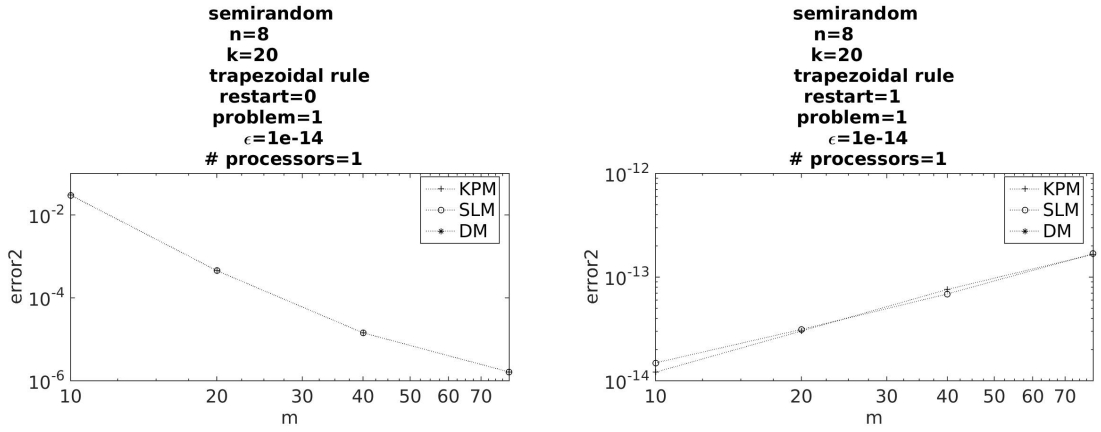
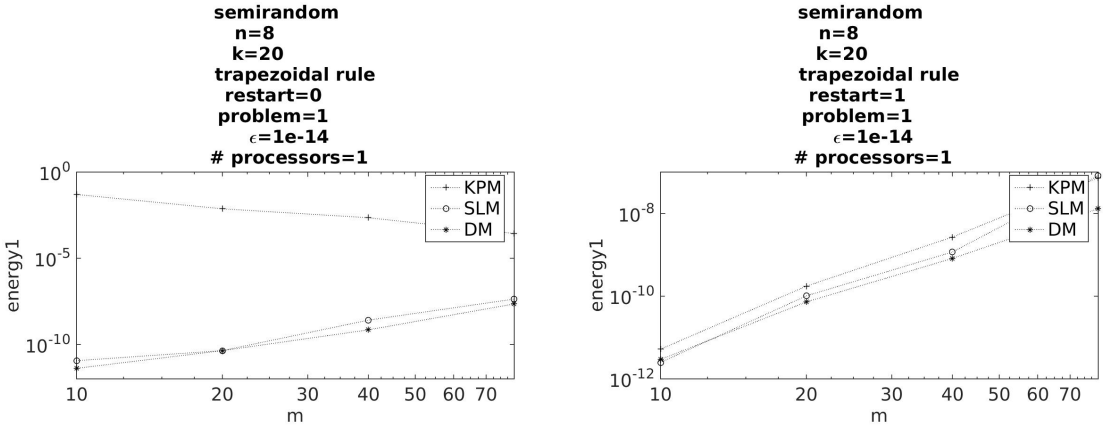


Figure 5.6: The number of iterations are almost equal for the two methods.



(a) (b)

Figure 5.7: The error is nearly identical.

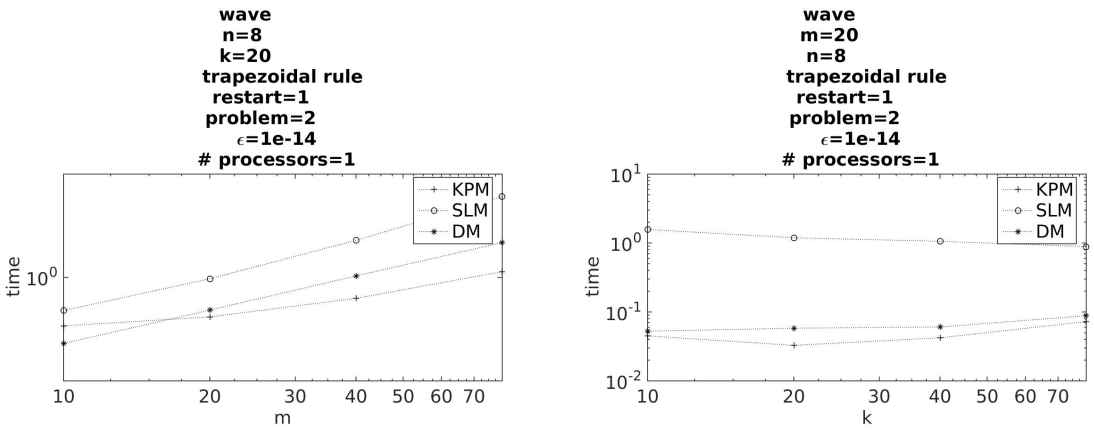


(a) (b)

Figure 5.8: SLM and KPM has the best energy estimation, with DM not far behind.

The figures in this section shows a little better how the energy for the different methods change, but the error is apparently useless in this setting due to it being the difference between two calculated numbers, and not the analytical solution.

5.3 Varying energy with the wave equation



(a) (b)

Figure 5.9: For both cases KPM is faster than the other, and with DM faster than SLM.

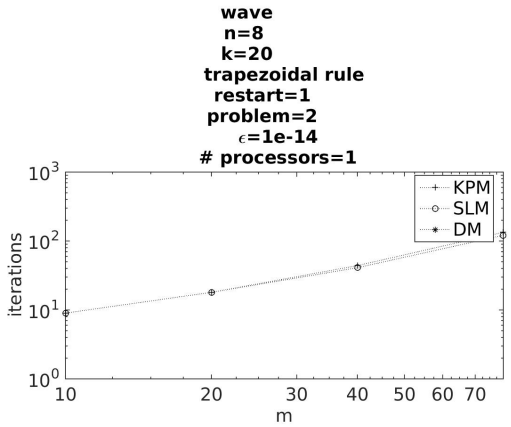
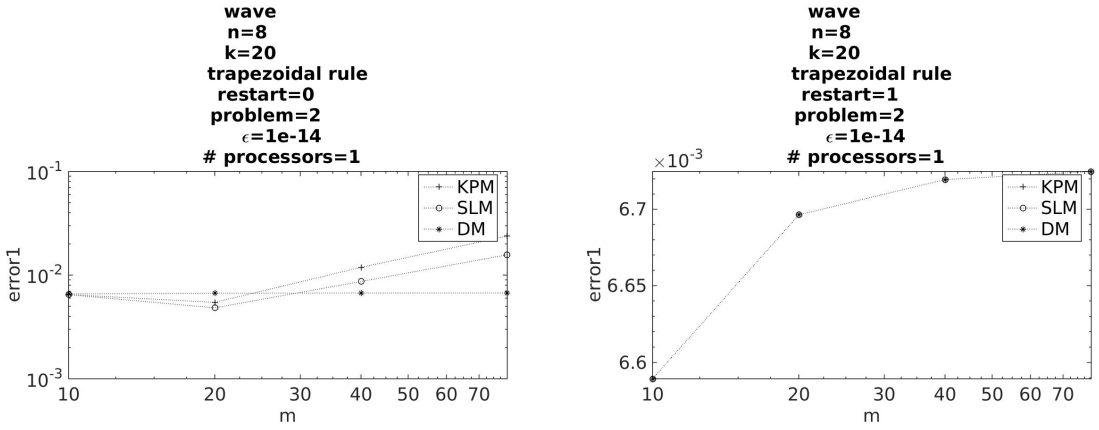
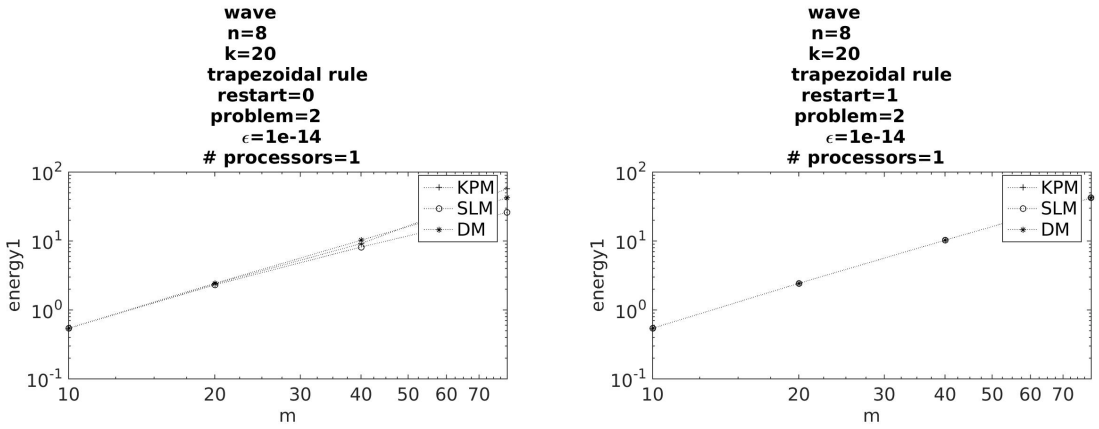


Figure 5.10: The number of iterations are almost equal for the two methods.



(a) (b)

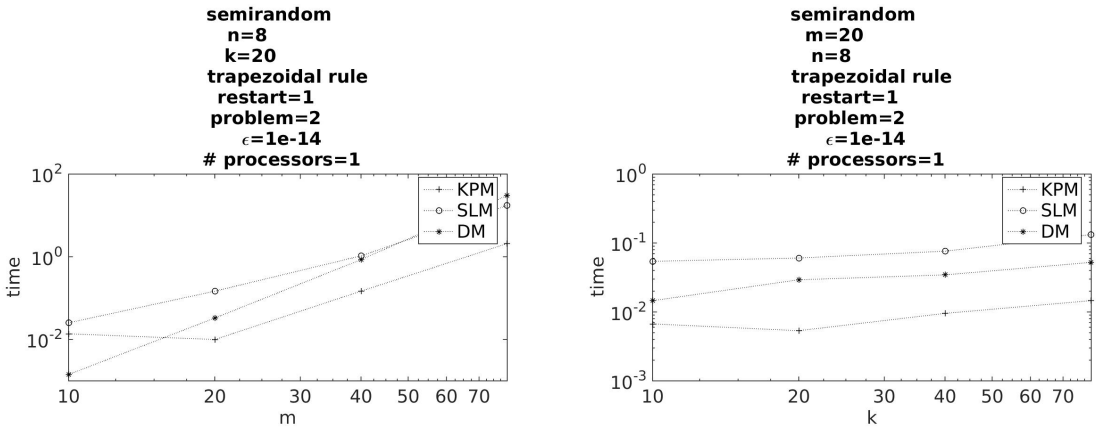
Figure 5.11: The error is nearly identical.



(a) (b)

Figure 5.12: SLM and KPM has the best energy estimation, with DM not far behind.

5.4 Varying energy with the semirandom equation



(a) (b)

Figure 5.13: For both cases KPM is faster than the other, and with DM faster than SLM.

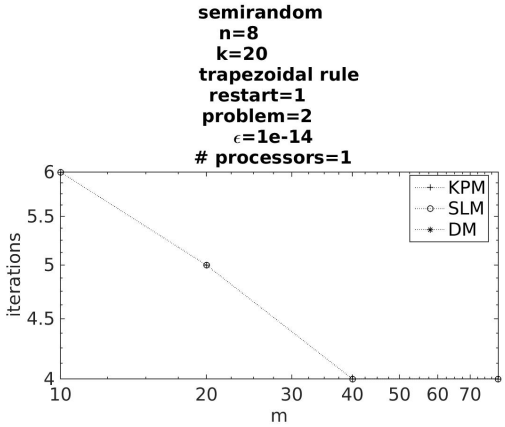
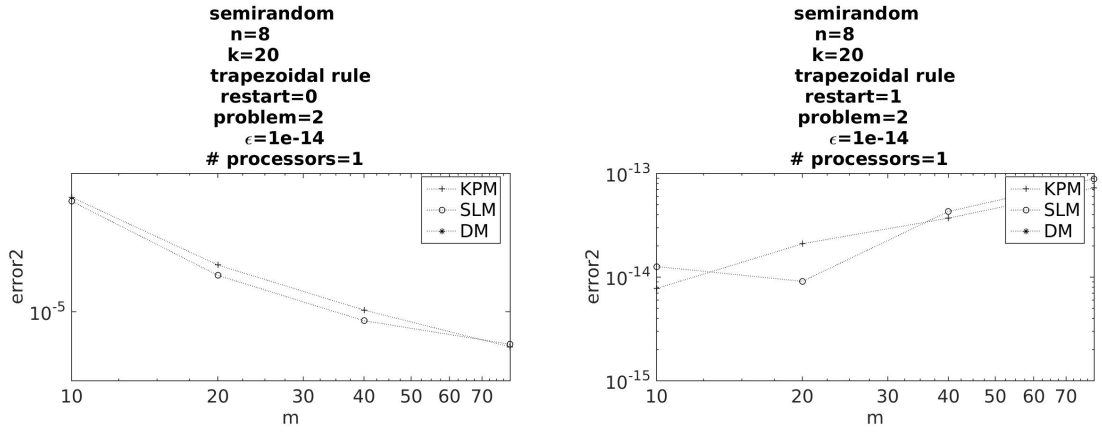
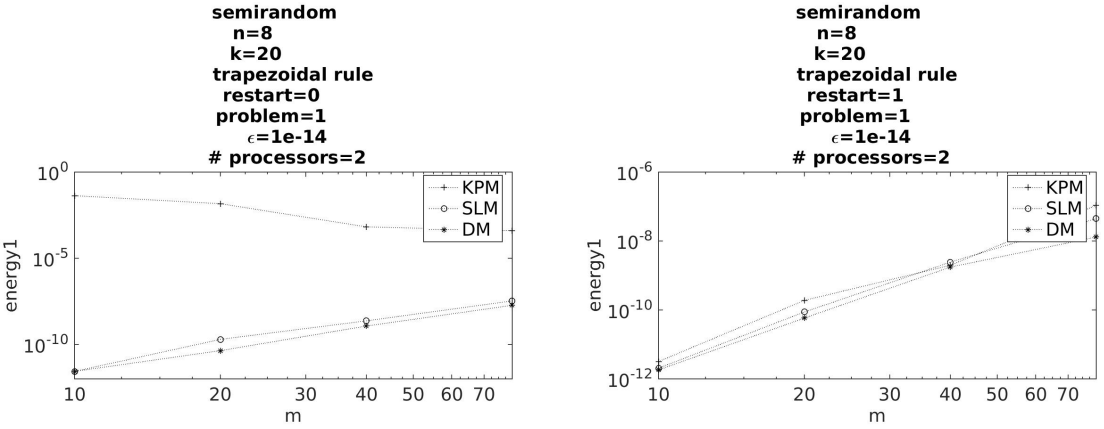


Figure 5.14: The number of iterations are almost equal for the two methods.



(a) (b)

Figure 5.15: The error is nearly identical.



(a) (b)

Figure 5.16: SLM and KPM has the best energy estimation, with DM not far behind.

It is clear that the restart improves the energy estimates for

Bibliography

[1] Celledoni E. and Moret I. A Krylov projection method for system of ODEs. *Applied Numerical Mathematics 23 (1997) 365-378*, 1997.

[2] Sindre Eskeland. A Krylov projection Method for the heat equation.

[3] Lu Li. SYMPLECTIC LANCOZS METHOD FOR SOLVING HAMILTONIAN SYSTEMS.

[4] Abramowitz Milton and Stegun Irene A. Handbook of Mathematical Functions. Tenth Printing, December 1972. with corrections.

[5] Heike Faßbender Peter Benner. An Implicitly Restarted SymplecUc Lanczos Method for the Hamlltonlan Eigenvalue Problem. *LINEAR ALGEBRA AND ITS APPLICATIONS 263:75-111*, 1997.

[6] Heike Faßbender Martin Stoll Peter Benner. A Hamiltonian Krylov–Schur-type method based on the symplectic Lanczos process. *Linear Algebra and its Applications*, (435):578–600, 2011.

[7] Endre Süli and David F. Mayers. *An Introduction to Numerical Analysis*, chapter 7.2 Newton–Cotes formulae, pages 202–203. cambridge university press, 2003.

[8] Endre Süli and David F. Mayers. *An Introduction to Numerical Analysis*, chapter 12.2 One-step methods, page 317. cambridge university press, 2003.

[9] Endre Süli and David F. Mayers. *An Introduction to Numerical Analysis*, chapter Definition 10.1, page 286. cambridge university press, 2003.

[10] Saad Yousef. *Iterative Methods for Sparse Linear Systems*, volume SECOND EDITION, chapter 6.3.1, page 154. Siam, 2003. Algorithm 6.1.