Solving ill-conditioned linear algebraic systems by the dynamical systems method (DSM)

N. S. Hoang* and A. G. Ramm†
Mathematics Department, Kansas State University, Manhattan, KS 66506, USA
email: {nguyenhs, ramm}@math.ksu.edu
March 24, 2008

Abstract
An iterative scheme for the Dynamical Systems Method (DSM) is given such that one does not have to solve the Cauchy problem occurring in the application of the DSM for solving ill-conditioned linear algebraic systems. The novelty of the algorithm is that the algorithm does not have to find the regularization parameter $a$ by solving a nonlinear equation. Numerical experiments show that DSM competes favorably with the Variational Regularization.

Keywords: ill-posed problems, ill-conditioned linear algebraic systems, dynamical systems method (DSM).

AMS subject classification: 65F10, 65F22.

1 Introduction
Consider a linear operator equation of the form

$$F(u) = Au - f = 0, \quad u \in H,$$

where $H$ is a Hilbert space and $A$ is a linear operator in $H$ which is not necessarily bounded but closed and densely defined. To solve this equation we apply the Dynamical Systems Method (DSM) introduced in [6]:

$$u' = -u + (T + a(s))^{-1}A^*f, \quad u(0) = u_0,$$

where $T := A^*A$ and $a(t) > 0$ is a nonincreasing function such that $a(t) \to 0$ as $t \to \infty$. The unique solution to (2) is given by

$$u(t) = u_0e^{-t} + e^{-t}\int_0^t e^s(T + a(s))^{-1}A^*f ds.$$
The DSM consists of solving problem (2) with a chosen \( a(t) \) and \( u_0 \) and finding a stopping time \( t_\delta \) so that \( u(t_\delta) \) approximates the solution \( y \) to problem (1) of minimal norm. Different choices of \( a(t) \) generate different methods of solving equation (1). These methods have different accuracy and different computation time. Thus, in order to get an efficient implementation of the DSM, we need to study the choice of \( a(t) \) and of the stopping time \( t_\delta \). Since the solution to (1) can be presented in the form of an integral, the question arises: how can one compute the integral efficiently? The integrand of the solution is used also in the Variational Regularization (VR) method. The choice of the stopping time \( t_\delta \) will be done by a discrepancy-type principle for DSM. However, choosing \( a(t) \) so that the method will be accurate and the computation time is small is not a trivial task.

This paper deals with the following questions:

1. How can one choose \( a(t) \) so that the DSM is fast and accurate?
2. Does the DSM compete favorably with the VR in computation time?
3. Is the DSM comparable with the VR in accuracy?

2 Construction of method

2.1 An iterative scheme

Let us discuss a choice of \( a(t) \) which allows one to solve problem (2) or to calculate the integral (3) without using any numerical method for solving initial-value problem for ordinary differential equations (ODE). In fact, using a monotonically decreasing \( a(t) \) with one of the best numerical methods for nonstiff ODE, such as DOPRI45, is more expensive computationally than using a step function \( \tilde{a}(t) \), approximating \( a(t) \), but brings no improvement in the accuracy of the solution to our problems compared to the numerical solution of our problems given in Section 3.1.2.

Necessary conditions for the function \( a(t) \) are: \( a(s) \) is a nonincreasing function and \( \lim_{s \to \infty} a(s) = 0 \) (see [6]). Thus, our choice of \( a(t) \) must satisfy these conditions. Consider a step function \( \tilde{a}(t) \), approximating \( a(t) \), defined as follows:

\[
\tilde{a}(t) = a_n, \quad t_n \leq t < t_{n+1},
\]

the number \( t_n \) are chosen later. For this \( \tilde{a}(t) \), \( u_n = u(t_n) \) can be computed
by the formula:

\[ u_n = u_0 e^{-t_n} + e^{-t_n} \sum_{i=1}^{n} (e^{t_i} - e^{t_{i-1}})(T + a_{i-1})^{-1} A^* f_\delta. \]

This leads to the following iterative formula:

\[ u_{n+1} = e^{-h_n} u_n + (1 - e^{-h_n})(T + a_n)^{-1} A^* f_\delta, \quad h_n = t_{n+1} - t_n. \]  

Thus, \( u_n \) can be obtained iteratively if \( u_0, a(t) \) and \( t_n \) are known.

The questions are:

1. For a given \( a(t) \), how can we choose \( t_n \) or \( h_n \) so that the DSM works efficiently?

2. With \( a_n = a(t_n) \) where \( a(t) \) is a continuous function, does the iterative scheme compete favorably with the DSM version in which \( u(t) \) is solved by some numerical methods such as Runge-Kutta methods using \( a(t) \)?

In our experiments, \( a_n = a(t_n) \) where \( a(t) = \frac{a_0}{1 + t} \) where \( a_0 > 0 \) is a constant which will be chosen later, as suggested in [6], with \( t_n \) chosen so that \( t_{n+1} - t_n = h_n, \ h_n = q^n, \) where \( 1 \leq q \leq 2 \). For this choice, if \( q > 1 \) then the solution \( u_n \) at the \( n \)-th step depends mainly on \( (T + a_n)^{-1} A^* f_\delta \) since \( e^{-h_n} \) is very small when \( n \) is large.

Note that \( a_n \) decays exponentially fast when \( n \to \infty \) if \( q > 1 \). A question arises: how does one choose \( q \) so that the method is fast and accurate? This question will be discussed in Section 3.

ALGORITHM 1 demonstrates the use of the iterative formula (4) and a relaxed discrepancy principle described below for finding \( u \) given \( a_0, A, f_\delta \) and \( \delta \).

In order to improve the speed of the algorithm, we use a relaxed discrepancy principle: at each iteration one checks if

\[ 0.9\delta \leq ||Au_n - f_\delta|| \leq 1.001\delta. \]  

As we shall see later, \( a_0 \) is chosen so that the condition (7) (see below) is satisfied. Thus, if \( u_0 = T^{-1} a^* f_\delta \), where \( T_a := T + a \), then \( \delta < ||Au_0 - f_\delta|| \). Let \( t_n \) be the first time such that \( ||Au_n - f_\delta|| \leq 1.001\delta \). If (6) is satisfied, then one stops calculations. If \( ||Au_n - f_\delta|| < 0.9\delta \), then one takes a smaller step-size and recomputes \( u_n \). If this happens, we do not increase \( h_n \), that is, we do not multiply \( h_n \) by \( q \) in the following steps. One repeats this procedure until condition (6) is satisfied.
Algorithm 1: DSM($A, f_\delta, \delta$)

$q := 2$;
$g_\delta := A^*f_\delta$;  $T := A^*A$;
$\text{itermax} := 30$;  $u = (T + a_0)^{-1}g_\delta$;
$i := 0$;  $t = 1$;  $h := 1$;  $\text{halve} := 0$;

while $(1.001\delta < \|Au - f_\delta\|)$ and $(i < \text{itermax})$ do

  $i := i + 1$;  $t = t + h$;  $a = a_0/t$;
  $v := (T + aI)^{-1}g_\delta$;
  $\tilde{u} = e^{-h}u + (1 - e^{-h})v$;
  if $0.9\delta < \|A\tilde{u} - f_\delta\|$ then
    $u := \tilde{u}$;
    if $\text{halve} = 0$ then $h := hq$;  end;
  elseif $t := t - h$;  $h := h/2$;  $\text{halve} = 1$;
  endif
endwhile

In order to improve the speed of the algorithm, we use a relaxed discrepancy principle: at each iteration one checks if

$$0.9\delta \leq \|Au_n - f_\delta\| \leq 1.001\delta.$$  \hfill (6)

As we shall see later, $a_0$ is chosen so that the condition (7) (see below) is satisfied. Thus, if $u_0 = T_{a_0}^{-1}A^*f_\delta$, where $T_a := T + a$, then $\delta < \|Au_0 - f_\delta\|$. Let $t_n$ be the first time such that $\|Au_n - f_\delta\| \leq 1.001\delta$. If (8) is satisfied, then one stops calculations. If $\|Au_n - f_\delta\| < 0.9\delta$, then one takes a smaller step-size and recomputes $u_n$. If this happens, we do not increase $h_n$, that is, we do not multiply $h_n$ by $q$ in the following steps. One repeats this procedure until condition (8) is satisfied.

2.2 On the choice of $a_0$

From numerical experiments with ill-conditioned linear algebraic systems (las) of the form $Au = f_\delta$, it follows that the regularization parameter $a_M$, obtained from the discrepancy principle $\|Au_{a_M} - f_\delta\| = \delta$, where $u_{a_M} = T_{a_M}^{-1}A^*f_\delta$, is often close to the optimal value $a_{op}$, i.e., the value minimizing the quantity:

$$\|u_{a_{op}} - y\| = \inf_a \|u_a - y\|, \quad u_a = T_a^{-1}Af_\delta.$$
The letter $M$ in $a_M$ stands for Morozov, who suggested to choose $c = 1$ in the discrepancy principle.

If $a_0$ is chosen smaller than $a_{op}$, the method may converge poorly. Since $a_M$ is close to $a_{op}$, only those $a$ for which $\|AT_a^{-1}A^*f_\delta - f_\delta\| = c\delta$ with $c$ ‘close’ to 1 yield accurate approximations to the solution $y$. Also, if $a_0$ is chosen much greater than $a_{op}$, then the information obtained from the starting steps of the iterative process (11) is not valuable because when $a_0$ is far from $a_{op}$, the error $\|u_{a_0} - y\|$ is much bigger than $\|u_{a_{op}} - y\|$. If $a_0$ is much bigger than $a_{op}$, a lot of time will be spent until $a(t_n)$ becomes close to $a_{op}$. In order to increase the speed of computation, $a_0$ should be chosen so that it is close to $a_{op}$ and greater than $a_{op}$. Since $a_{op}$ is not known and is often close to $a_M$, we choose $a_0$ from the condition:

$$\delta < \|Au_{a_0} - f_\delta\| < 2\delta.$$ (7)

For this choice, $a_0$ is ‘close’ to and greater than $a_M$. Since there are many $a_0$ satisfying this condition, it is not difficult to find one of them.

In the implementation of the VR using discrepancy principle with Morozov’s suggestion $c = 1$, if one wants to use the Newton method for finding the regularization parameter, one also has to choose the starting value $a_0$ so that the iteration process converges, because the Newton method, in general, converges only locally. If this value is close to and greater than $a_M$, it can also be used as the initial value of $a_0 = a(t)|_{t=0}$ for the DSM.

In our numerical experiments, with a guess $a_0 = \frac{1}{3} \max \lambda_i(A^*A)\delta_{rel}$ for $a(0)$, we find $a_0$ such that $\delta < \|Au_{a_0} - f_\delta\| < 2\delta$. Here, $\delta_{rel}$ stands for the relative error, i.e., $\delta_{rel} = \frac{\delta}{\|f\|}$. The factor $\frac{1}{3}$ is introduced here in order to reduce the cost for finding $a_0$, because $a_0$, which satisfies (7), is often less than $\max \lambda_i(A^*A)\delta_{rel}$. The idea for this choice is based on the fact that the spectrum of the matrix $\frac{1}{\max \lambda_i(A^*A)}A^*A$ is contained in $[0, 1]$.

Note that one has

$$a_M \leq \frac{\delta\|A\|^2}{\|f_\delta\| - \delta}.$$ 

Indeed,

$$\|f_\delta\| - \delta = \|f_\delta\| - \|Au_{a_M} - f_\delta\| \leq \|Au_{a_M}\|.$$ 

Since $A^*Au_{a_M} + a_Mu_{a_M} = A^*f_\delta$, one has $a_MAu_{a_M} = AA^*(f_\delta - Au_{a_M})$. Thus,

$$\|f_\delta\| - \delta \leq \|Au_{a_M}\| = \frac{1}{a_M}\|AA^*(f_\delta - Au_{a_M})\| \leq \frac{\|A\|^2}{a_M}\delta.$$
Similar estimate one can find in [5, p. 53], where \( a_0 = \frac{\delta \| A \|^2}{\| f \| - \delta} \) is suggested as a starting value for Newton’s method to determine \( a_M \) on the basis that it is an upper bound for \( a_M \). Note that \( \frac{\delta \| A \|^2}{\| f \| - \delta} \approx \delta_{rel} \| A \|^2 = \max \lambda_i (A^* A) \delta_{rel} \). However, in practice Newton’s method does not necessarily converge with this starting value. If this happens, a smaller starting value \( a_1 := \frac{a_0}{2} \) is used to restart the Newton’s method.

In general, our initial choice for \( a_0 \) may not satisfy (7). Iterations for finding \( a_0 \) to satisfy (7) are done as follows:

1. If \( \frac{\| Au_0 - f \delta \|}{\delta} = c > 3 \), then one takes \( a_1 := \frac{a_0}{2(c-1)} \) as the next guess and checks if the condition (7) is satisfied. If \( 2 < c \leq 3 \) then one takes \( a_1 := a_0/3 \).

2. If \( \frac{\| Au_0 - f \delta \|}{\delta} = c < 1 \), then \( a_1 := 3a_0 \) is used as the next guess.

3. After \( a_0 \) is updated, one checks if (7) is satisfied. If (7) is not satisfied, one repeats steps 1 and 2 until one finds \( a_0 \) satisfying condition (7) (see ALGORITHM 2).

**ALGORITHM 2: find-\( a_0 \)**

\[
\begin{align*}
a_0 &:= \frac{1}{2} \| A \|^2 \delta_{rel} ; \\
c &:= \| Au_{a_0} - f \delta \| / \delta ; \\
&\textbf{while} (2 < c) \textbf{or} (c < 1) \textbf{do} \\
&\text{if } 3 < c \text{ then} \\
&\quad a_0 := 0.5a_0/(c - 1) ; \\
&\text{elseif } 2 < c \leq 3 \text{ then} \\
&\quad a_0 := a_0/3 ; \\
&\text{else} \\
&\quad a_0 := 3a_0 ; \\
&\text{end} \\
&\quad u_{a_0} := (A^* A + a_0)^{-1} A^* f \delta ; \\
&\quad c := \| Au_{a_0} - f \delta \| / \delta ; \\
&\textbf{endwhile}
\end{align*}
\]

The above strategy is based on the fact that the function

\[
\phi(a) = \| A(T + a)^{-1} A^* f \delta - f \delta \|
\]
is a monotonically decreasing function of \( a \), \( a > 0 \). In looking for \( a_0 \), satisfying (7), when our guess \( a_0 \gg a_M > 0 \) or \( \|Au_{a_0} - f\delta\| \gg \delta \), one uses an approximation

\[
\phi(x) \approx \phi(a_0) + (x - a_0) \frac{\phi(a_0) - \phi(a_M)}{a_0 - a_M}.
\]

Note that \( \phi(a_0) \) and \( a_0 \) are known. We are looking for \( x \) such that \( \delta < \varphi(x) < 2\delta \). Thus, if \( a_1 \) is such that \( \delta < \varphi(a_1) < 2\delta \) and if \( 2\delta < \phi(a_0) \), then

\[
(\phi(a_0) - 2\delta)\frac{a_0}{\phi(a_0) - \delta} < a_0 - a_1 < (\phi(a_0) - \delta)\frac{a_0}{\phi(a_0) - \delta}.
\]

Hence, we choose \( a_1 \) such that

\[
a_0 - a_1 = (\phi(a_0) - 1.5\delta)\frac{a_0}{\phi(a_0) - \delta},
\]

so

\[
a_1 = a_0 \frac{0.5\delta}{\phi(a_0) - \delta}.
\]

Although this is a very rough approximation, it works well in practice. It often takes 1 to 3 steps to get an \( a_0 \) satisfying (7). That is why we have a factor \( \frac{0.5\delta}{\phi(a_0) - \delta} \) in the first case. Overall, it is easier to look for \( a_0 \) satisfying (7) than to look for \( a_0 \) for which the Newton’s method converges. Indeed, the Newton’s scheme for solving \( a_M \) does not necessarily converge with \( a_0 \) found from condition (7).

3 Numerical experiments

In this section, we compare DSM with VR\(_i\) and VR\(_n\). In all methods, we begin with the guess \( a_0 = \frac{1}{\|A\|} \|A\|^2 \delta_{rel} \) and use the ALGORITHM 2 to find \( a_0 \) satisfying condition (7). In our experiments, the computation cost for this step is very low. Indeed, it only takes 1 or 2 iterations to get \( a_0 \). By VR\(_i\) we denote the VR obtained by using \( a = a_0 \), the initial value for \( a(t) \) in DSM, and by VR\(_n\) we denote the VR with \( a = a_M \), found from the VR discrepancy principle with \( c = 1 \) by using Quasi-Newton’s method with the initial guess \( a = a_0 \). Quasi-Newton’s method is chosen instead of Newton’s method in order to reduce the computation cost. In all experiments we compare these methods in accuracy and with respect to the parameter \( N_{insol} \), which is the
number of times for solving the linear system $T_a u = A^* f_\delta$ for $u$. Note that solving these linear systems is the main cost in these methods.

In this section, besides comparing the DSM with the VR for linear algebraic systems with Hilbert matrices, we also carry out experiments with other linear algebraic systems given in the Regularization package in [4]. These linear systems are obtained as a part of numerical solutions to some integral equations. Here, we only focus on the numerical methods for solving linear algebraic systems, not on solving these integral equations. Therefore, we use these linear algebraic systems to test our methods for solving stably these systems.

### 3.1 Linear algebraic systems with Hilbert matrices

Consider a linear algebraic system

$$H_n u = f_\delta, \quad (8)$$

where

$$f_\delta = f + e, \quad f = H_n x, \quad H_n = \begin{bmatrix} 1 & \frac{1}{2} & \cdots & \frac{1}{n} \\ \frac{1}{2} & \frac{1}{3} & \cdots & \frac{1}{n+1} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{n} & \frac{1}{n+1} & \cdots & \frac{1}{2n-1} \end{bmatrix},$$

and $e \in \mathbb{R}^n$ is a random normally distributed vector such that $\|e\|_2 \leq \delta_{rel}\|f\|_2$. The Hilbert matrix $H_n$ is well-known for having a very large condition number when $n$ is large. If $n$ is sufficiently large, the system is severely ill-conditioned.

#### 3.1.1 The condition numbers of Hilbert matrices

It is impossible to calculate the condition number of $H_n$ by computing the ratio of the largest and the smallest eigenvalues of $H_n$ because for large $n$ the smallest eigenvalue of $H_n$ is smaller than $10^{-16}$. Note that singular values of $H_n$ are its eigenvalues since $H_n$ is selfadjoint and positive definite. Due to the limitation of machine precision, every value smaller than $10^{-16}$ is understood as 0. That is why if we use the function `cond` provided by MATLAB, the condition number of $H_n$ for $n \geq 20$ is about $10^{16} \times \max |\lambda_i(H_n)|$. Since the largest eigenvalue of $H_n$ grows very slowly, the condition numbers of $H_n$ for $n \geq 20$ are all about $10^{20}$, while, in fact, the condition number of $H_{100}$ computed by the formula, given below, is about $10^{150}$ (see Table 1).
general, computing condition numbers of strongly ill-conditioned matrices is an open problem. The function \texttt{cond}, provided by MATLAB, is not always reliable for computing the condition number of ill-condition matrices. Fortunately, there is an analytic formula for the inverse of $H_n$. Indeed, one has (see [2])

$$H_n^{-1} = (h_{ij})_{i,j=1}^n,$$

where

$$h_{ij} = (-1)^{i+j} (i+j-1) \left( \frac{n+i-1}{n-j} \right) \left( \frac{n+j-1}{n-i} \right) \left( \frac{i+j-2}{i-1} \right)^2.$$

Thus, the condition number of the Hilbert matrix can be computed by the formula:

$$\text{cond}(H_n) = \|H_n\| \|H_n^{-1}\|.$$

Here \text{cond}(H_n) stands for the condition number of the Hilbert matrix $H_n$ and $\|H_n\|$ and $\|H_n^{-1}\|$ are the largest eigenvalues of $H_n$ and $H_n^{-1}$, respectively. Although MATLAB can not compute values less than $10^{-16}$, it can compute values up to $10^{200}$. Therefore, it can compute $\|H_n^{-1}\|$ for $n$ up to 120. In MATLAB, the matrices $H_n$ and $H_n^{-1}$ can be obtained by the syntax: $H_n = \text{hilb}(n)$ and $H_n^{-1} = \text{invhilb}(n)$, respectively.

The condition numbers of Hilbert matrices, computed by the above formula, are given in Table 1.

| $n$ | 20 | 40 | 60 | 80 | 100 | 120 |
|-----|----|----|----|----|-----|-----|
| \text{cond}(H_n) | $2.5 \times 10^{28}$ | $7.7 \times 10^{38}$ | $2.7 \times 10^{58}$ | $9.9 \times 10^{78}$ | $3.8 \times 10^{98}$ | $1.5 \times 10^{118}$ |

From Table 1 one can see that the computed condition numbers of the Hilbert matrix grow very fast as $n$ grows.

### 3.1.2 Continuous $a(t)$ compared to the step function $\tilde{a}(t)$

In this section, we compare the DSM, which is implemented by solving the Cauchy problem (2) with $a(t)$, and the iterative DSM implemented with $\tilde{a}(t)$ approximating $a(t)$ as described in Section 2.1. Both of them use the same $a_0$ which is found by ALGORITHM 2. The DSM using a numerical method to solve the Cauchy problem is implemented as follows:

1. One uses the DOPRI45 method which is an embedded pair consisting of a Runge-Kutta (RK) method of order 5 and another RK method of order 4 which is used to estimate the error in order to control the step sizes. The DOPRI45 is an explicit method which requires 6 right-hand
side function evaluations at each step. Details about the coefficients and variable step size strategy can be found in [1, 3]. Using a variable step size helps to choose the best step sizes and improves the speed.

2. In solving (2), at the end of each step, one always checks the stopping rule, based on the discrepancy principle

\[0.9 \leq \|Au_\delta(t) - f_\delta\| \leq 1.001\delta.\]

If this condition is satisfied, one stops and takes the solution at the final step \(u(t_n)\) as the solution to the linear algebraic system.

Table 2: Numerical results for Hilbert matrices for \(\delta_{rel} = 0.01, n = 100.\)

| \(n\) | DSM \(N_{insol} \frac{\|u_\delta - y\|_2}{\|y\|_2}\) | DSM \((q = 1)\) \(N_{insol} \frac{\|u_\delta - y\|_2}{\|y\|_2}\) | DSM-DOPRI45 \(N_{insol} \frac{\|u_\delta - y\|_2}{\|y\|_2}\) |
|---|---|---|---|
| 10 | 5 | 0.1222 | 10 | 0.1195 | 205 | 0.1223 |
| 20 | 5 | 0.1373 | 7 | 0.1537 | 145 | 0.1584 |
| 30 | 6 | 0.0945 | 20 | 0.1180 | 313 | 0.1197 |
| 40 | 5 | 0.2174 | 7 | 0.2278 | 151 | 0.2290 |
| 50 | 6 | 0.1620 | 14 | 0.1609 | 247 | 0.1609 |
| 60 | 6 | 0.1456 | 16 | 0.1478 | 253 | 0.1480 |
| 70 | 6 | 0.1436 | 13 | 0.1543 | 229 | 0.1554 |
| 80 | 6 | 0.1778 | 10 | 0.1969 | 181 | 0.1963 |
| 90 | 6 | 0.1531 | 13 | 0.1535 | 307 | 0.1547 |
| 100 | 7 | 0.1400 | 23 | 0.1522 | 355 | 0.1481 |

The DSM version implemented with the DOPRI45 method is denoted DSM-DOPRI45 while the other iterative version of DSM is denoted just by DSM.

Table 2 presents the numerical results with Hilbert matrices \(H_n\) obtained by two versions of the DSM for \(n = 10, 20, ..., 100, \delta_{rel} = 0.01, x = (x_1, ..., x_n)^T, x_i = \sqrt{\frac{2i-1}{100}}\pi.\) From Table 2, as well as other numerical experiments, we found that the accuracy obtained by the DSM-DOPRI45 is worse than that of the iterative DSM. Moreover, the computation time for the DSM-DOPRI45 is much greater than that for the iterative DSM. Also, using \(h = \text{const}\) or \(q = 1\) does not give more accurate solutions while requires more computation time.

The conclusion from this experiment as well as from other experiments is that the DSM with \(q = 2\) is much faster and often gives better results than the DSM with \(q = 1\) and the DSM-DOPRI45. Therefore, we choose the iterative DSM with \(q = 2\) to compare with the VR\(_n\) method.
3.1.3 DSM compared to VR

In this section, we test three methods: the DSM, the VR$_i$ and the VR$_n$ on linear algebraic systems with Hilbert matrices. The first linear system is obtained by taking $H_{100}$ and $x = (x_1, \ldots, x_{100})^T$, where $x_i = (i-1)^2$. For the second problem we just change $x_i$ to $x_i = \sin(2\frac{i-1}{100}\pi)$. Numerical results for these systems are shown in Figure 1.

$$H_{100}$$

![Figure 1](https://example.com/figure1.png)

Figure 1: Plots of solutions obtained by the DSM and VR with the exact solution $x$, $x = (x_i)_{i=1}^{100}$ when $x_i = \left(2\frac{i-1}{100}\pi\right)^2$ (left) and $x_i = \sin\left(2\frac{i-1}{100}\pi\right)$ (right) with $\delta_{rel} = 0.02$.

Looking at Figure 1 one can see that with the same guess $a_0$, both the VR$_n$ and DSM give better results than those of VR$_i$. As it can be seen from Figure 1, the numerical solutions obtained by the DSM in these tests are slightly more accurate than those of the VR$_n$.

Table 3 presents results with Hilbert matrices $H_n$ for $n = 10, 20, \ldots, 100$, $\delta_{rel} = 0.01$, $x = (x_1, \ldots, x_n)^T$, $x_i = \sqrt{2\frac{i-1}{100}\pi}$. Looking at this Table it is clear that the results obtained by the DSM are slightly more accurate than those by the VR$_n$ even in the cases when the VR$_n$ requires much more work than the DSM. In this example, we can conclude that the DSM is better than the VR$_n$ in both accuracy and time of computation.
3.2 A linear algebraic system related to an inverse problem for the heat equation

In this section, we apply the DSM and the VR to solve a linear algebraic system used in the test problem heat from Regularization tools in [4]. This linear algebraic system is a part of numerical solutions to an inverse problem for the heat equation. This problem is reduced to a Volterra integral equation of the first kind with \([0, 1]\) as the integration interval. The kernel is \(K(s, t) = k(s - t)\) with

\[
k(t) = \frac{t^{-3/2}}{2\kappa\sqrt{\pi}} \exp\left(-\frac{1}{4\kappa^2 t}\right).
\]

Here, we use the default value \(\kappa = 1\). In this test in [4] the integral equation is discretized by means of simple collocation and the midpoint rule with \(n\) points. The unique exact solution \(u_n\) is constructed, and then the right-hand side \(b_n\) is produced as \(b_n = A_n u_n\) (see [4]). In our test, we use \(n = 10, 20, ..., 100\) and \(b_{n,\delta} = b_n + e_n\), where \(e_n\) is a vector containing random entries, normally distributed with mean 0, variance 1, and scaled so that \(\|e_n\| = \delta_{rel}\|b_n\|\). This linear system is ill-posed: the condition number of \(A_{100}\) obtained by using the function cond provided in MATLAB is \(1.3717 \times 10^{37}\). As we have discussed earlier, this condition number may be not accurate because of the limitations of the program cond provided in MATLAB. However, this number shows that the corresponding linear algebraic system is ill-conditioned.

Looking at the Table 4 one can see that in some situations the VR\(_n\) is not
as accurate as the DSM even when it takes more iterations than the DSM. Overall, the results obtained by the DSM are often slightly more accurate than those by the VR\(_n\). The time of computation of the DSM is comparable to that of the VR\(_n\). In some situations, the results by VR\(_n\) and the VR\(_i\) are the same although it uses 3 more iterations than does the DSM. The conclusion from this Table is that DSM competes favorably with the VR\(_n\) in both accuracy and time of computation.

Figure 2 plots numerical solutions to the inverse heat equation for \(\delta_{rel} = 0.05\) and \(\delta_{rel} = 0.02\) when \(n = 100\). From the figure we can see that the numerical solutions obtained by the DSM are about the same those by the VR\(_n\). In these examples, the time of computation of the DSM is about the same as that of the VR\(_n\).

The conclusion is that the DSM competes favorably with the VR\(_n\) in this experiment.

### 3.3 A linear algebraic system for the computation of the second derivatives

Let us do some numerical experiments with linear algebraic systems arising in a numerical experiment of computing the second derivative of a noisy function.

The problem is reduced to an integral equation of the first kind. A linear algebraic system is obtained by a discretization of the integral equation.
The inverse heat equation with $n=100$, $\delta_{rel}=0.05$ and $\delta_{rel}=0.1$.

Figure 2: Plots of solutions obtained by DSM, VR for the inverse heat equation when $n = 100$, $\delta_{rel} = 0.05$ (left) and $\delta_{rel} = 0.1$ (right).

whose kernel $K$ is Green’s function

$$K(s, t) = \begin{cases} s(t - 1), & \text{if } s < t \\ t(s - 1), & \text{if } s \geq t \end{cases}$$

Here $s, t \in [0, 1]$ and as the right-hand side $f$ and the corresponding solution $u$ one chooses one of the following (see [4]):

- case 1, $f(s) = (s^3 - s)/6$, $u(t) = t$,
- case 2, $f(s) = e^s + (1 - e)s - 1$, $u(t) = e^t$,
- case 3, $f(s) = \begin{cases} (4s^3 - 3s)/24, & \text{if } s < \frac{1}{2} \\ (-4s^3 + 12s^2 - 9s + 1)/24, & \text{if } s \geq \frac{1}{2} \end{cases}$, $u(t) = \begin{cases} t, & \text{if } t < \frac{1}{2} \\ 1 - t, & \text{if } t \geq \frac{1}{2} \end{cases}$.

Using $A_n$ and $u_n$ in [4], the right-hand side $b_n = A_n u_n$ is computed. Again, we use $n = 10, 20, \ldots, 100$ and $b_{n, \delta} = b_n + e_n$, where $e_n$ is a vector containing random entries, normally distributed with mean 0, variance 1, and scaled so that $\|e_n\| = \delta_{rel} \|b_n\|$. This linear algebraic system is mildly ill-posed: the condition number of $A_{100}$ is $1.2158 \times 10^4$.

Numerical results for the third case is presented in Table 5. In this case, the results obtained by the VR$_n$ are often slightly more accurate than
those of the DSM. However, the difference between accuracy as well as the difference between time of computation of these methods is small. Numerical results obtained by these two methods are much better than those of the VR$_i$.

Table 5: Results for the deriv2 problem with $\delta_{rel} = 0.01$, $n = 100$ case 3.

| $n$ | $N_{\text{limsol}}$ | $\|u_d - y\|_2/\|y\|_2$ | $N_{\text{limsol}}$ | $\|u_d - y\|_2/\|y\|_2$ | $N_{\text{limsol}}$ | $\|u_d - y\|_2/\|y\|_2$ |
|-----|---------------------|--------------------------|---------------------|--------------------------|---------------------|--------------------------|
| 10  | 4                   | 0.0500                   | 2                   | 0.0542                   | 6                   | 0.0444                   |
| 20  | 4                   | 0.0584                   | 2                   | 0.0708                   | 6                   | 0.0561                   |
| 30  | 4                   | 0.0690                   | 2                   | 0.0718                   | 6                   | 0.0661                   |
| 40  | 4                   | 0.0367                   | 1                   | 0.0454                   | 4                   | 0.0384                   |
| 50  | 3                   | 0.0564                   | 1                   | 0.0565                   | 4                   | 0.0562                   |
| 60  | 4                   | 0.0426                   | 1                   | 0.0452                   | 4                   | 0.0407                   |
| 70  | 5                   | 0.0499                   | 1                   | 0.0422                   | 5                   | 0.0372                   |
| 80  | 4                   | 0.0523                   | 1                   | 0.0516                   | 4                   | 0.0498                   |
| 90  | 4                   | 0.0446                   | 1                   | 0.0493                   | 4                   | 0.0456                   |
| 100 | 4                   | 0.0399                   | 1                   | 0.0415                   | 5                   | 0.0391                   |

For other cases, case 1 and case 2, numerical results obtained by the DSM are slightly more accurate than those by the VR$_i$. Figure 3 plots the numerical solutions for these cases. The computation time of the DSM in these cases is about the same as or less than that of the VR$_n$.

Figure 3: Plots of solutions obtained by DSM, VR for the deriv2 problem when $n = 100$, $\delta_{rel} = 0.02$ (left) and $\delta_{rel} = 0.02$ (right).
The conclusion in this experiment is that the DSM competes favorably with the VR. Indeed, the VR is slightly better than the DSM in case 3 but slightly worse than the DSM in cases 1 and 2.

4 Concluding remarks

The conclusions from the above experiments are:

1. The DSM always converges for $a(t) = \frac{a_0}{1+t}$ given that $a_0 > a_{op}$. However, if $a_0$ is not well chosen, then the convergence speed may be slow. The parameter $a_0$ should be chosen so that it is greater than and close to the optimal $a_{op}$, i.e., the value minimizing the quantity:

$$\|u_{a_{op}} - y\| = \inf_a \|u_a - y\|, \quad u_a = T_a^{-1} A f_\delta.$$

However, since $a_{op}$ is not known and $a_M$ is often close to $a_{op}$, we choose $a_0$ so that

$$\delta < \|AT_{a_0}^{-1} A^* f_\delta - f_\delta\| < 2\delta.$$

2. The DSM is sometimes faster than the VR. In general, the DSM is comparable with the VR with respect to computation time.

3. The DSM is often slightly more accurate than the VR, especially when $\delta$ is large. Starting with $a_0$ such that $\delta < \|AT_{a_0}^{-1} A^* f_\delta - f_\delta\| < 2\delta$, the DSM often requires 4 to 7 iterations, and main cost in each iteration consists of solving the linear system $T_a u = A^* f_\delta$. The cost of these iterations is often about the same as the cost of using Newton’s method to solve $a_M$ in the VR.

4. For any initial $a_0$ such that $\delta < \|AT_{a_0}^{-1} A^* f_\delta - f_\delta\| < 2\delta$, the DSM always converges to a solution which is often more accurate than that of the VR. However, with the same initial $a_0$, the VR does not necessarily converge. In this case, we restart the Newton scheme to solve for the regularization parameter with initial guess $a_1 = \frac{a_0}{2}$ instead of $a_0$.

References

[1] Burrage, K., 1995, Parallel and sequential methods for ordinary differential equations, Oxford University Press, Oxford.
[2] Choi, M. D., 1983, Tricks or treats with the Hilbert matrix, American Mathematical Monthly, pp. 301-312.

[3] Hairer, E. and Nørsett, S.P. and Wanner, G., 1987, Solving ordinary differential equation I, nonstiff problems, Springer, Berlin.

[4] Hansen, P. C., 1994, Regularization tools, A Matlab package for the analysis and solution of discrete ill-posed problems. Numerical Algorithms, pp. 1-35.

[5] Kirsch, A., 1986, An introduction to the mathematical theory of inverse problems, Springer-Verlag, Berlin.

[6] Ramm, A. G., 2007, Dynamical systems method for solving operator equations, Elsevier, Amsterdam.