SIAM J. MATRIX ANAL. APPL. © 2000 Society for Industrial and Applied Mathematics
Vol. 22, No. 1, pp. 145–154

RATIONAL MATRIX FUNCTIONS AND RANK-1 UPDATES*

DANIEL S. BERNSTEIN† AND CHARLES F. VAN LOAN‡

Abstract. Suppose $f = p/q$ is a quotient of two polynomials and that p has degree r_p and q has degree r_q. Assume that $f(A)$ and $f(A + uv^T)$ are defined where $A \in \mathbb{R}^{n\times n}$, $u \in \mathbb{R}^n$, and $v \in \mathbb{R}^n$ are given and set $r = \max\{r_p, r_q\}$. We show how to compute $f(A + uv^T)$ in $O(rn^2)$ flops assuming that $f(A)$ is available together with an appropriate factorization of the "denominator matrix" $q(A)$. The central result can be interpreted as a generalization of the well-known Sherman–Morrison formula. For an application we consider a Jacobian computation that arises in an inverse problem involving the matrix exponential. With certain assumptions the work required to set up the Jacobian matrix can be reduced by an order of magnitude by making effective use of the rank-1 update formulae developed in this paper.

Key words. matrix functions, Sherman–Morrison

AMS subject classifications. 65F05, 65F99, 65L05

PII. S0895479898333636

1. Introduction.

Suppose $A \in \mathbb{R}^{n\times n}$, $u \in \mathbb{R}^n$, and $v \in \mathbb{R}^n$ are given and that f is a prescribed rational function that is defined on the spectrum of A and $A+uv^T$. In this paper we are concerned with the efficient computation of $f(A + uv^T)$ assuming that $f(A)$ is available.

The special case $f(z) = 1/z$ is well known:

$$(A + uv^T)^{-1} = A^{-1} - \alpha y z^T, \qquad y = A^{-1}u,\ z = A^{-T}v,\ \alpha = 1/(1 + z^T u).$$

This is the *Sherman–Morrison formula* and it can be used to compute $(A + uv^T)^{-1}$ from A^{-1} in $O(n^2)$ flops. See [2, p. 50]. In a linear equation setting, the Sherman–Morrison result, together with a QR factorization of A, makes it possible to solve $(A + uv^T)x = b$ in $O(n^2)$ flops.

Interest in a "fast" $f(A + uv^T)$ result can arise in several settings. For example, Benzi and Golub [1] present a Lanczos-based process for bounding certain matrix functions. Our generalized Sherman–Morrison result widens the class of problems that can be efficiently handled by their technique. Kenney and Laub [3] discuss condition estimation for matrix functions. With our results it is possible to compute more specialized sensitivity measures, e.g., how a rational function of a matrix A changes when a particular a_{ij} is varied.

The paper is organized as follows. In the next section we derive the generalized Sherman–Morrison formula and a closed-form expression for the derivative of $f(A)$ with respect to a_{ij}. Numerical results, stability issues, and a Jacobian evaluation application are discussed in section 3.

2. A generalized Sherman–Morrison result.

We first show that if A changes by a rank-1 matrix, then A^j changes by a rank-j matrix. Krylov matrices and exchange permutation matrices are involved. For $A \in \mathbb{R}^{n\times n}$, $x \in \mathbb{R}^n$, and $j > 0$, the

*Received by the editors February 3, 1998; accepted for publication (in revised form) by R. Freund October 28, 1999; published electronically May 31, 2000.
http://www.siam.org/journals/simax/22-1/33363.html

†Department of Computer Science, University of Massachusetts, 140 Governor's Drive, Amherst, MA 01003 (bern@cs.umass.edu).

‡Department of Computer Science, Cornell University, 4130 Upson Hall, Ithaca, NY 14853 (cv@cs.cornell.edu).

Krylov matrix $\mathrm{Kry}(A,x,j) \in \mathbb{R}^{n\times j}$ is defined by

$$\mathrm{Kry}(A,x,j) = \big[x,\ Ax, \ldots, A^{j-1}x\big] \in \mathbb{R}^{n\times j}.$$

We adopt the convention that $\mathrm{Kry}(A,x,j)$ is the empty matrix if $j=0$. The *exchange permutation matrix* $E_j \in \mathbb{R}^{j\times j}$ is just the identity matrix I_j with its columns in reverse order, i.e.,

$$E_j = I_j(:, j{:}-1{:}1)\,.$$

Lemma 1. *If $A \in \mathbb{R}^{n\times n}$, $u \in \mathbb{R}^n$, $v \in \mathbb{R}^n$, and $j > 0$, then*

$$(A+uv^T)^j \;=\; A^j + K_j E_j L_j^T,$$

where $K_j = \mathrm{Kry}(A,u,j)$ and $L_j = \mathrm{Kry}(A^T+vu^T, v, j)$.

Proof. We use induction. The lemma is true for $j=1$ since $K_1 = u$, $E_1 = I_1$, and $L_1 = v$. Assume that the lemma holds for some $j \geq 1$. It follows that

$$\begin{aligned}
(A+uv^T)^{j+1} &= (A+uv^T)^j(A+uv^T) \;=\; (A^j + K_jE_jL_j^T)(A+uv^T)\\
&= A^{j+1} + K_jE_jL_j^T(A+uv^T) + A^juv^T\\
&= A^{j+1} + K_j((A^T+vu^T)L_jE_j)^T + (A^ju)v^T\\
&= A^{j+1} + K_j\left[\,(A^T+vu^T)^jv\,, \ldots,\ (A^T+vu^T)v\,\right]^T \;+\; (A^ju)v^T\\
&= A^{j+1} + \big[K_j\ ,\ A^ju\big]\begin{bmatrix} v^T(A+uv^T)^j \\ \vdots \\ v^T(A+uv^T) \\ v^T \end{bmatrix}\\
&= A^{j+1} + K_{j+1}E_{j+1}L_{j+1}^T. \qquad \Box
\end{aligned}$$

The computation of $(A+uv^T)^j$ from A^j requires approximately $6jn^2$ flops if the lemma is carefully exploited.

The next result shows that if A changes by a rank-1 matrix, then a degree-r polynomial in A changes by a rank-r matrix. *Hankel* matrices are involved and for $\alpha = (\alpha_1, \ldots, \alpha_r)$ we define

$$\mathrm{Hank}(\alpha) \;=\; \begin{bmatrix} \alpha_1 & \alpha_2 & \alpha_3 & \cdots & \alpha_r \\ \alpha_2 & \alpha_3 & \cdots & \cdots & 0 \\ \alpha_3 & \vdots & \ddots & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \alpha_r & 0 & 0 & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{r\times r}.$$

Lemma 2. *If $A \in \mathbb{R}^{n\times n}$, $u \in \mathbb{R}^n$, $v \in \mathbb{R}^n$, and $p(z) = \alpha_0 + \alpha_1 z + \cdots + \alpha_r z^r$ with $r > 0$, then*

$$p(A+uv^T) \;=\; p(A) + K_rH_\alpha L_r^T,$$

where $K_r = \mathrm{Kry}(A,u,r)$, $L_r = \mathrm{Kry}(A^T+vu^T, v, r)$, $\alpha = (\alpha_1, \ldots, \alpha_r)$, and $H_\alpha = \mathrm{Hank}(\alpha)$.

Proof. Using Lemma 1 we have

$$p(A+uv^T) = \sum_{j=0}^{r} \alpha_j (A+uv^T)^j \;=\; \alpha_0 I \;+\; \sum_{j=1}^{r} \alpha_j (A^j + K_j E_j L_j^T)$$
$$= p(A) + \sum_{j=1}^{r} \alpha_j K_j E_j L_j^T,$$

where $K_j = \mathrm{Kry}(A,u,j)$, $L_j = \mathrm{Kry}(A^T + vu^T, v, j)$, and E_j is the j-by-j exchange permutation. Note that K_j and L_j are the first j columns of $K_r = \mathrm{Kry}(A,u,r)$ and $L_r = \mathrm{Kry}(A^T+vu^T, v, r)$, i.e., $K_j = K_r(:,1{:}j)$ and $L_j = L_r(:,1{:}j)$. Let `zeros(m,n)` denote the m-by-n zeros matrix as in MATLAB with the convention that it is the empty matrix if either argument is zero. It follows that

$$K_j E_j L_j^T = K_r \begin{bmatrix} E_j & 0 \\ 0 & \texttt{zeros}(r-j, r-j) \end{bmatrix} L_r^T$$

and so

$$p(A+uv^T) = p(A) \;+\; K_r \left(\sum_{j=1}^{r} \alpha_j \begin{bmatrix} E_j & 0 \\ 0 & \texttt{zeros}(r-j, r-j) \end{bmatrix} \right) L_r^T \; .$$

This proves the lemma since the matrix inside the parentheses is precisely the Hankel matrix H_α defined above. □

The computation of $p(A+uv^T)$ from $p(A)$ involves about $6n^2r + nr^2$ flops if the lemma is carefully exploited.

We are now set to prove that if A changes by a rank-1 matrix, then a rational function of A changes by a rank-r matrix where r is the maximum degree of the numerator and denominator polynomials.

THEOREM 3. *Suppose $f(z) = p(z)/q(z)$, where*

$$p(z) \;=\; \sum_{i=0}^{r_p} \alpha_i z^i \qquad \textit{and} \qquad q(z) \;=\; \sum_{i=0}^{r_q} \beta_i z^i.$$

Let $r = \max\{r_p, r_q\}$ and define the r-vectors

$$\tilde{\alpha} \;=\; (\,\alpha_1, \ldots, \alpha_{r_p}, \underbrace{0, \ldots, 0}_{r-r_p}) \qquad \textit{and} \qquad \tilde{\beta} \;=\; (\,\beta_1, \ldots, \beta_{r_q}, \underbrace{0, \ldots, 0}_{r-r_q}).$$

Suppose $A \in \mathbb{R}^{n\times n}$, $u \in \mathbb{R}^n$, $v \in \mathbb{R}^n$ and that $f(A)$ and $f(A+uv^T)$ are defined. Set $H_{\tilde{\alpha}} = \mathrm{Hank}(\tilde{\alpha})$ and $H_{\tilde{\beta}} = \mathrm{Hank}(\tilde{\beta})$. If

$$K_r = \mathrm{Kry}(A,u,r), \tag{1}$$
$$L_r = \mathrm{Kry}(A^T + vu^T, v, r), \tag{2}$$
$$Y_\alpha^T = H_{\tilde{\alpha}} L_r^T, \tag{3}$$
$$Y_\beta^T = H_{\tilde{\beta}} L_r^T, \tag{4}$$

then

$$f(A+uv^T) \;=\; f(A) + XY^T, \tag{5}$$

where

$$X = q(A)^{-1}K_r, \tag{6}$$
$$Y^T = Y_\alpha^T - M^{-1}Y_\beta^T(f(A) + XY_\alpha^T), \tag{7}$$

where

$$M = I_r + Y_\beta^T X. \tag{8}$$

Proof. Assume for clarity that both r_p and r_q are positive. The proof is easily adapted to handle the cases $r_p = 0$ and/or $r_q = 0$. (Various matrices and vectors below turn out to be empty.)

Set $\alpha = (\alpha_1, \ldots, \alpha_{r_p})$ and $\beta = (\beta_1, \ldots, \beta_{r_q})$ and define $H_\alpha = \text{Hank}(\alpha)$ and $H_\beta = \text{Hank}(\beta)$. Noting that $K_{r_p} = K_r(:, 1{:}r_p)$, $L_{r_p} = L_r(:, 1{:}r_p)$, $K_{r_q} = K_r(:, 1{:}r_q)$, and $L_{r_q} = L_r(:, 1{:}r_q)$, we see from Lemma 2 that

$$\begin{aligned} p(A + uv^T) &= P + K_{r_p}H_\alpha L_{r_p}^T \;=\; P + K_r H_{\tilde\alpha} L_r^T, \\ q(A + uv^T) &= Q + K_{r_q}H_\beta L_{r_q}^T \;=\; Q + K_r H_{\tilde\beta} L_r^T, \end{aligned}$$

where $P = p(A)$ and $Q = q(A)$. Thus,

$$\begin{aligned} f(A + uv^T) &= q(A + uv^T)^{-1}p(A + uv^T) \\ &= \left(Q + K_r H_{\tilde\beta} L_r^T\right)^{-1}\left(P + K_r H_{\tilde\alpha} L_r^T\right) \\ &= \left(I_n + Q^{-1}K_r H_{\tilde\beta} L_r^T\right)^{-1} Q^{-1}\left(P + K_r H_{\tilde\alpha} L_r^T\right) \\ &= \left(I_n + XY_\beta^T\right)^{-1}\left(F + XY_\alpha^T\right), \end{aligned}$$

where $F = f(A) = Q^{-1}P$. By the Sherman–Morrison–Woodbury formula [2, p. 50],

$$\left(I_n + XY_\beta^T\right)^{-1} \;=\; I_n - XM^{-1}Y_\beta^T,$$

where $M = I_r + Y_\beta^T X$ and so

$$\begin{aligned} f(A + uv^T) &= \left(I_n - XM^{-1}Y_\beta^T\right)\left(F + XY_\alpha^T\right) \\ &= F + X\left(Y_\alpha^T - M^{-1}Y_\beta^T(F + XY_\alpha^T)\right) \\ &= F + X\left(Y_\alpha^T - (I_r + Y_\beta^T X)^{-1}Y_\beta^T(F + XY_\alpha^T)\right) \\ &= F + XY^T. \qquad \square \end{aligned}$$

The computation of $f(A + uv^T)$ from $f(A)$ via (1)–(4) and (6)–(8) requires $\mathrm{O}(n^2 r)$ flops. Indeed, if we assume that $r = r_p = r_q$ and that a QR factorization of the denominator polynomial $q(A)$ is available, then the work is distributed as follows:

Calculation	Flops
K_r	$2n^2r$
L_r	$2n^2r$
Y_α	nr^2
Y_β	nr^2
X	$3n^2r$
M	$2nr^2$
Y	$4n^2r + 2nr^2$

This totals to $11n^2r$ flops if we assume that $r \ll n$. As with any Sherman–Morrison-type computation, the condition numbers of $q(A)$ and the matrix M defined by (8) should be monitored because they shed light on the accuracy of the computed update.

Note that Theorem 3 can be generalized in the direction of the Sherman–Morrison–Woodbury formula (see [2, p. 50]). In particular, if UV^T is a rank-d matrix and $f(A+UV^T)$ exists, then it can be shown that $f(A)$ and $f(A+UV^T)$ differ by a matrix that has rank dr.

We conclude this section by using Theorem 3 to develop an expression for the partial derivative of $f(A)$ with respect to a particular matrix element a_{ij}, i.e.,

$$\frac{\partial}{\partial a_{ij}} f(A) \;=\; \lim_{\delta \to 0} \frac{f(A+\delta e_i e_j^T) - f(A)}{\delta},$$

where $I_n = [\, e_1, \ldots, e_n \,]$. The idea is to use Theorem 3 to simplify the difference between $f(A)$ and $f(A+\delta e_i e_j^T)$.

COROLLARY 4. *Suppose $f(z) = p(z)/q(z)$, where*

$$p(z) \;=\; \sum_{i=0}^{r_p} \alpha_i z^i \qquad \textit{and} \qquad q(z) \;=\; \sum_{i=0}^{r_q} \beta_i z^i.$$

Let $r = \max\{r_p, r_q\}$ and define the r-vectors

$$\tilde{\alpha} \;=\; (\,\alpha_1, \ldots, \alpha_{r_p}, \underbrace{0, \ldots, 0}_{r-r_p}\,) \qquad \textit{and} \qquad \tilde{\beta} \;=\; (\,\beta_1, \ldots, \beta_{r_q}, \underbrace{0, \ldots, 0}_{r-r_q}\,).$$

Assume that $A \in \mathbb{R}^{n\times n}$, $f(A)$ is defined, and $1 \le i, j \le n$. If $H_{\tilde{\alpha}} = \mathrm{Hank}(\tilde{\alpha})$, $H_{\tilde{\beta}} = \mathrm{Hank}(\tilde{\beta})$, $Q = q(A)$, $K^{(i)} = \mathrm{Kry}(A, e_i, r)$, and $L^{(j)} = \mathrm{Kry}(A^T, e_j, r)$, then

$$\frac{\partial}{\partial a_{ij}} f(A) \;=\; X^{(i)} Z^{(j)T},$$

where

$$X^{(i)} = Q^{-1} K^{(i)}, \tag{9}$$

$$Z^{(j)T} = H_{\tilde{\alpha}} L^{(j)T} - H_{\tilde{\beta}} L^{(j)T} f(A). \tag{10}$$

Proof. Since $f(A)$ is defined, then the matrix $f(A+\delta e_i e_j^T)$ is also defined for all δ less than or equal to some sufficiently small δ_0. Assuming that $\delta \le \delta_0$, we apply Theorem 3 with $u = e_i$ and $v = \delta e_j$. It follows that if

$$\begin{aligned} L(\delta) &= \mathrm{Kry}(A^T + \delta e_j e_i^T, \delta e_j, r), \\ Y_\alpha(\delta)^T &= H_{\tilde{\alpha}} L(\delta)^T, \\ Y_\beta(\delta)^T &= H_{\tilde{\beta}} L(\delta)^T, \end{aligned}$$

then

$$f(A+\delta e_i e_j^T) = f(A) + X^{(i)} Y(\delta)^T,$$

where

$$Y(\delta)^T \;=\; Y_\alpha(\delta)^T - (I_r + Y_\beta(\delta)^T X^{(i)})^{-1} Y_\beta(\delta)^T (f(A) + X^{(i)} Y_\alpha(\delta)^T).$$

Since

$$L(\delta) \;=\; \delta \cdot \mathrm{Kry}(A^T + \delta e_j e_i^T, e_j, r),$$

we see that

$$\lim_{\delta \to 0} \frac{Y_\alpha(\delta)}{\delta} = H_{\tilde{\alpha}} L^{(j)T} \qquad \text{and} \qquad \lim_{\delta \to 0} \frac{Y_\beta(\delta)}{\delta} = H_{\tilde{\beta}} L^{(j)T}.$$

Since $Y_\alpha(\delta)$ and $Y_\beta(\delta)$ both converge to zero as $\delta \to 0$, it follows that

$$\lim_{\delta \to 0} Y(\delta)^T = \lim_{\delta \to 0} \left(\frac{Y_\alpha(\delta)^T}{\delta} \;-\; (I_r + Y_\beta(\delta)^T X^{(i)})^{-1} \frac{Y_\beta(\delta)^T}{\delta} (f(A) + X^{(i)} Y_\alpha(\delta)^T) \right)$$

$$= H_{\tilde{\alpha}} L^{(j)T} - H_{\tilde{\beta}} L^{(j)T} f(A) \;\equiv\; Z^{(j)T}. \qquad \square$$

Note that if f is a polynomial, then $H_{\tilde{\beta}} = 0$.

3. Discussion. We have written a MATLAB function

```
[XP,YP,XQ,YQ,XF,YF,condM] = Rational_Update(A,u,v,alfa,P,beta,Q_cell)
```

that implements the update formulae derived in the proof of Theorem 3. The vectors `alfa` and `beta` contain the coefficients of the numerator and denominator polynomials p and q, respectively. The QR factorization of $Q = q(A)$ is passed via a cell array representation `Q_cell`. If `A` is n-by-n and r is the larger of deg(p) and deg(q), then the output matrices `XP`, `YP`, `XQ`, `YQ`, `XF`, and `YF` are n-by-r and relate $p(A)$ to $p(A + uv^T)$, $q(A)$ to $q(A + uv^T)$, and $f(A)$ to $f(A + uv^T)$ as follows:

$$\begin{aligned} p(A + uv^T) &= P + X_P Y_P^T, \\ q(A + uv^T) &= Q + X_Q Y_Q^T, \\ f(A + uv^T) &= F + X_F Y_F^T. \end{aligned}$$

The r-by-r matrix M defined by (8) has an important role to play in the calculation of Y_F, and so its condition number is returned.

Table 1 reports on the quality of $f(A + uv^T)$ when f is the diagonal Pade approximation to the exponential function:

$$f(A) = \left(\sum_{\mu=0}^{r} \beta_\mu A^\mu \right)^{-1} \left(\sum_{\mu=0}^{r} \alpha_\mu A^\mu \right), \quad \alpha_\mu = \frac{(2r-\mu)!r!}{(2r)!\mu!(r-\mu)!}, \quad \beta_\mu = (-1)^\mu \alpha_\mu. \tag{11}$$

See [2, p. 572]. `Rational_Update` is used to update $f(A)$. The matrix A is a randomly generated 100-by-100 example with eigenvalues in the left-half plane and $\| A \|_2 = 15$. The denominator matrix $q(A)$ is well conditioned. The rank-1 correction uv^T has unit 2-norm in the test case. Define

$$\begin{aligned} \tilde{F}_0 &= \texttt{expm(A)}, \\ F_0 &= \text{the } (p,p) \text{ Pade approximation to } \exp(A), \\ \tilde{F}_1 &= \texttt{expm(A+u*v')}, \\ F_1 &= \text{the } (p,p) \text{ Pade approximation to } \exp(A + uv^T), \\ F_1^{(up)} &= \text{an estimate of } F_1 \text{ obtained by updating } F_0, \end{aligned}$$

TABLE 1
Errors associated with the update of the (r,r)-Pade approximation to e^A.

r	$\frac{\Vert F_0 - \tilde{F}_0 \Vert_2}{\Vert \tilde{F}_0 \Vert_2}$	$\frac{\Vert F_1 - \tilde{F}_1 \Vert_2}{\Vert \tilde{F}_1 \Vert_2}$	$\frac{\Vert F_1 - F_1^{(up)} \Vert_2}{\Vert F_1 \Vert_2}$	`cond(Q)`	`cond(M)`
0	2.00e+001	1.99e+001	0.00e+000	1.0e+000	1.0e+000
1	1.56e+001	1.55e+001	2.01e−015	3.6e+000	1.0e+000
2	9.42e+000	9.40e+000	3.17e−015	8.9e+000	1.0e+000
3	4.36e+000	4.35e+000	7.66e−015	1.7e+001	1.5e+000
4	1.52e+000	1.52e+000	2.31e−014	2.7e+001	4.7e+000
5	4.01e−001	4.03e−001	3.95e−014	3.9e+001	9.9e+000
6	8.08e−002	8.14e−002	4.46e−014	5.0e+001	3.5e+002
7	1.27e−002	1.28e−002	4.97e−014	6.2e+001	1.2e+004
8	1.59e−003	1.61e−003	5.35e−014	7.3e+001	2.7e+005
9	1.62e−004	1.64e−004	5.80e−014	8.3e+001	4.9e+006
10	1.37e−005	1.39e−005	6.15e−014	9.3e+001	7.9e+007
11	9.76e−007	9.88e−007	6.36e−015	1.0e+002	1.1e+009
12	5.94e−008	6.00e−008	6.88e−014	1.1e+002	1.5e+010
13	3.12e−009	3.14e−009	6.93e−014	1.2e+002	1.8e+011
14	1.43e−010	1.43e−010	7.46e−014	1.2e+002	2.0e+012
15	5.77e−012	5.75e−012	7.80e−014	1.3e+002	2.1e+013
16	2.06e−013	2.05e−013	7.89e−014	1.4e+002	2.0e+014

where `expm` is the built-in MATLAB function for the matrix exponential. In this study we treat `expm` as exact, thereby enabling us to report on the relative error in F_0 and F_1. We are well aware of the difficulties associated with e^A calculations, but this example is not numerically challenging for `expm`.

An interesting aspect of Table 1 is that the 2-norm relative error in $F_1^{(up)}$ is consistently small even as the condition of the matrix M deteriorates with increasing r. We have no analysis or informal explanation for this phenomena, which (it turns out) is quite typical. The Krylov matrices that "make up" the matrix M and the "right-hand-side matrix" $Y_\beta^T(f(A) + XY_\alpha^T)$ in (7) are notoriously ill conditioned, especially for large values of r. But somehow the effect of this ill-conditioning is muted. It is perhaps worth noting that Krylov matrices are always involved (at least implicitly) with *any* matrix polynomial calculation because

$$\sum_{i=0}^{r} \alpha_i A^i = [I, A, \ldots, A^r] \begin{bmatrix} \alpha_0 I \\ \alpha_1 I \\ \vdots \\ \alpha_r I \end{bmatrix}.$$

Apparently there is not a simple connection between the condition of a matrix polynomial computation and the condition of the Krylov matrices that lurk in the background. Clearly, more research in this area is required, especially if high-degree polynomials are involved.

We now proceed to a discussion of Corollary 4 and how it can be applied in a systems identification context. Suppose the value of a scalar-valued function

$$y(t) = c^T e^{At} b, \qquad b, c \in \mathbb{R}^n,\ A \in \mathbb{R}^{n\times n},$$

is known at $t = t_1, \ldots, t_m$ with $m \geq n^2$ and that from that data we want to reconstruct A. Defining

$$y_k = y(t_k), \qquad k = 1{:}m,$$

we see that y_k is a snapshot of the solution to the initial value problem

$$\dot{x} = Ax, \; x(0) = b$$

"as seen through" the observation vector c, i.e., $y_k = c^T x(t_k)$.

There are several ways to attack this difficult identification problem (see [4, 5]). One approach is to approximate e^{At_k} with a rational function $f_k(A)$ for $k = 1{:}m$ and then to minimize $\| \phi(A) - y \|_2$, where

$$\phi(A) = \begin{bmatrix} c^T f_1(A)b \\ c^T f_2(A)b \\ \vdots \\ c^T f_m(A)b \end{bmatrix} \qquad \text{and} \qquad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}.$$

This is a nonlinear least square problem in the n^2 entries that define the matrix $A = (a_{ij})$. If the Levenberg–Marquardt algorithm is applied, then (among other things) at each step we need to compute the m-by-n^2 Jacobian J of the vector-valued function $\phi(A)$.

Suppose an ODE solver is used to generate $f_k(A)b \approx x(t_k)$ for $k = 1{:}m$. If each of these vectors requires $\mathrm{O}(n^2)$ flops to compute, then a ϕ-evaluation costs $\mathrm{O}(mn^2)$ flops and a finite-difference approximation to J would involve $\mathrm{O}(mn^4)$ flops. We show how this flop count can be reduced by a factor that ranges from n to n^2.

Note that the kth row of the Jacobian involves partial derivatives of the scalar-valued function $c^T f_k(A)b$ with respect to each of the matrix entries a_{ij}:

$$\frac{\partial}{\partial a_{ij}} \left(c^T f_k(A) b \right) \; = \; c^T \left(\frac{\partial}{\partial a_{ij}} f_k(A) \right) b.$$

Corollary 4 is therefore applicable. Let us see how we might use the corollary to simplify the evaluation of these Jacobian entries. Assume that f_k is the (r_p, r_q) Pade approximation to e^{At_k}, i.e.,

$$f_k(A) = q_k(A)^{-1} p_k(A),$$

$$p_k(A) = \left(\sum_{\mu=0}^{r_p} \alpha_\mu^{(k)} A^\mu \right),$$
$$q_k(A) = \left(\sum_{\mu=0}^{r_q} \beta_\mu^{(k)} A^\mu \right)$$

with

$$\alpha_\mu^{(k)} = \frac{(r_p + r_q - \mu)! r_p!}{(r_p + r_q)! \mu! (r_p - \mu)!} t_k^\mu \quad \text{and} \quad \beta_\mu^{(k)} = \frac{(r_p + r_q - \mu)! r_q!}{(r_p + r_q)! \mu! (r_q - \mu)!} t_k^\mu.$$

Using Corollary 4,

$$c^T \left(\frac{\partial}{\partial a_{ij}} f_k(A) \right) b \; = \; c^T X_k^{(i)} Z_k^{(j)T} b, \tag{12}$$

where

$$X_k^{(i)} = q_k(A)^{-1}K^{(i)}, \tag{13}$$

$$Z_k^{(i)} = H_{\alpha^{(k)}}L^{(j)T} - H_{\beta^{(k)}}L^{(j)T}q_k(A)^{-1}p_k(A). \tag{14}$$

Note that if $r = \max\{r_p, r_q\}$ and we precompute and store the matrix powers $A^2, \ldots, A^r$, then the Krylov matrices $K^{(i)}$ and $L^{(j)}$ are easily retrieved and the matrices $p_k(A)$ and $q_k(A)$ can be set up in $O(n^2r)$ flops. For a given Jacobian row index k, we must compute the vectors $c_{k1}, \ldots, c_{kn} \in \mathbb{R}^r$ defined by

$$c_{ki}^T = c^Tq_k(A)^{-1}K^{(i)},\ i = 1{:}n, \tag{15}$$

and the vectors $b_{k1}, \ldots, b_{kn} \in \mathbb{R}^r$ defined by

$$b_{ki} = H_{\alpha^{(k)}}L^{(i)T}b - H_{\beta^{(k)}}L^{(i)T}q_k(A)^{-1}p_k(A)b,\ i = 1{:}n. \tag{16}$$

The kth row of the Jacobian is then made up of the inner products $c_{ki}^Tb_{kj}$, where i and j each range from 1 to n. Summarizing the overall Jacobian evaluation we get

```
Compute and save the matrix powers $A^2, \ldots, A^r$.
For $k = 1{:}m$
    Set up $q_k(A)$ and $p_k(A)$, and compute the QR factorization of the former.
    For $i = 1{:}n$
        Compute the vectors $c_{ki}$ and $b_{ki}$ defined by (15) and (16).
    end
    Establish the $k$th row of the Jacobian by computing the inner products
        $c_{ki}^Tb_{kj}$, $i = 1{:}n, j = 1{:}n$.
end
```

The powers of A cost $O(rn^3)$. As we mentioned above, once these powers are available, then the Krylov matrices $K^{(i)}$ and $L^{(j)}$ that figure in the computations are "free." For each k there is an $O(n^3)$ QR factorization. The vectors $c_{k1}, \ldots, c_{kn}$ and $b_{k1}, \ldots, b_{kn}$ can altogether be computed in $O(rn^2)$ flops. The n^2 inner products $c_{ki}^Tb_{kj}$ cost another $O(rn^2)$ flops. Thus, each row of the Jacobian requires $O(rn^2 + n^3)$ flops. The total Jacobian evaluation as outlined is therefore an $O(rmn^2 + mn^3)$ computation, a factor of n less than the ODE finite-difference approach.

We note that if the Pade approximation is a polynomial ($r_q = 0$), then a number of simplifications result:

```
Compute and save the matrix powers $A^2, \ldots, A^r$.
For $i = 1{:}n$
    Compute $c_i^T = c^TK^{(i)}$ and $b_i = L^{(i)T}b$.
end
For $k = 1{:}m$
    Compute the inner products $c_i^TH_{\alpha^{(k)}}b_j$, $i = 1{:}n, j = 1{:}n$.
end
```

The flop count now is just $O(rmn^2)$, which is less than the ODE finite-difference approach by a factor of n^2.

A numerical issue here concerns the size of r. As we mentioned earlier, the Krylov-related computations can be problematic if this parameter is too large. However, in some contexts it is possible to base the identification process on observations of $c^Te^{At}b$ at values $t = t_1, \ldots, t_m$ that are small in value. In this case a low-order Pade approximation to e^{At_k} can suffice.

MATLAB software for carrying out the updates discussed in this paper is available from http://www.cs.cornell.edu/cv.

Acknowledgments. The authors became interested in this problem as a result of discussions with Professor Martha Contreras of the Biometrics Unit at Cornell University. Mike Todd read an earlier draft of this paper, spotted some errors, and showed us that the exact Jacobian could be obtained in section 2.

REFERENCES

[1] M. Benzi and G.H. Golub, *Bounds for the entries of matrix functions with applications to preconditioning*, BIT, 39 (1999), pp. 417–438.
[2] G.H. Golub and C.F. Van Loan, *Matrix Computations*, 3rd ed., The Johns Hopkins University Press, Baltimore, MD, 1997.
[3] C.S. Kenney and A.J. Laub, *Condition estimates for matrix functions*, SIAM J. Matrix Anal. Appl., 10 (1989), pp. 191–209.
[4] R.I. Jennrich and P.B. Bright, *Fitting systems of linear differential equations using computer generated exact derivatives*, Technometrics, 18 (1976), pp. 385–392.
[5] L. Ljung, *System Identification—Theory for the User*, Prentice-Hall, Englewood Cliffs, NJ, 1984.