Efficient Algorithms For Control System Design Using Computer Algebra

by

Robert Lawrence Munach

Paper submitted to the Department of Mechanical Engineering
Graduate School in partial fulfillment of the
requirements for the degree of
Master of Science
1989

Advisor: Professor John S. Baras
Department of Electrical Engineering
Director, Systems Research Center

Introduction

The theory of polynomial matrices plays a key role in the design of multi-input multi-output control and communication systems. Examples include coprime factorizations of transfer function matrices, canonical realizations obtained from matrix fraction descriptions, design of feedback compensators and convolutional coders. Typically, such problems abstract in a natural way to the need to solve systems of Diophantine Equations (e.g., the so-called Bezout Equation). These and other problems involving polynomial matrices require efficient polynomial matrix triangularization procedures, a result which is not surprising given the importance of matrix triangularization techniques in numerical linear algebra. There, matrices with entries from a field can be triangularized using Gaussian elimination. However, polynomial matrices have entries from a Euclidean ring, an algebraic object for which Gaussian elimination is not defined. In a Euclidean ring, triangularization is accomplished instead by Euclidean elimination. Unfortunately, the numerical stability and sensitivity issues of Euclidean elimination are not well understood, and a reliable numerical algorithm for this procedure does not exist at present.

This paper presents new algorithms which entirely circumvent these numerical sensitivity issues through the use of exact, symbolic methods in computer algebra. The use of such error-free algorithms guarantees that all calculations are exact and therefore accurate to within the precision of the model data — the best that can be hoped. The emphasis will be placed on efficient algorithms to compute *exact* Hermite forms of polynomial matrices because this procedure is central to a large variety of algorithms important in the design of control and communication systems. Moreover, the triangular Hermite form is defined for any matrix with entries from a principal ideal ring. Such matrices arise in many practical problems in communications and control, for instance, the analysis of quantization effects in linear systems and the design of convolutional coders. Due to their symbolic nature, the algorithms apply equally well to such problems since the particular ring involved in a particular problem can be itself considered as program input data.

We have implemented algorithms to compute the exact Hermite forms of polynomial matrices in the MACSYMA and *Mathematica* computer algebra languages. A suite of auxiliary programs which we have implemented will be discussed which call on these tri-

angularization procedures in order to perform more high-level tasks arising in the control system design process.

Facts And Terminology of Polynomials and Polynomial Matrices

Denote by $Q[s]$ the ring of polynomials in the indeterminant 's' with coefficients drawn from the field of rational numbers, Q. $Z[s]$ is the subring of $Q[s]$ formed when the polynomial coefficients are restricted to lie in Z, the ring of integers.

The *leading coefficient* of a polynomial is the nonzero coefficient of its highest degree term. By convention, the leading coefficient of the zero polynomial is defined to be one, and its degree is taken to be $-\infty$. If the leading coefficient of a polynomial is one, then the polynomial is said to be *monic*. If $a(s) \in Q[s]$ is a polynomial of degree zero, then $a(s)$ is a *unit* of $Q[s]$; i.e., $\frac{1}{a(s)} \in Q[s]$. A polynomial $p(s) \in Z[s]$ is called *primitive* if its coefficients are relatively prime in Z, i.e., if the greatest common divisor of its coefficients is a unit of Z, namely ± 1. For any $p(s) \in Z[s]$, there exists a non-zero scalar $c \in Z$, unique up to a unit in Z, and a primative polynomial $pp(s) \in Z[s]$, such that $p(s) = c\ pp(s)$; c is called the *content* of $p(s)$ and $pp(s)$ is its *primative*. A collection of polynomials in $Z[s]$ having contents which are relatively prime is said to be *relatively* primitive.

A polynomial $p(s)$ *divides* a polynomial $q(s)$, $p(s)|q(s)$, if there exists $c(s)$ such that $p(s)\ c(s) = q(s)$. A *common divisor*, CD, $c(s)$ of $\{p_i(s) : i = 1, \ldots, n\}$ is a polynomial such that $c(s)|\{p_i(s)\}$. A *greatest common divisor*, GCD, $g(s)$ of $\{p_i(s)\}$ is a CD of $\{p_i(s)\}$ such that $c(s)|g(s)$ for any other CD, $c(s)$, of $\{p_i(s)\}$.

A *common multiple*, CM, $c(s)$ of $\{p_i(s)\}$ is a polynomial such that $\{p_i(s)\}|c(s)$. A *least common multiple*, LCM, $l(s)$ of $\{p_i(s)\}$ is a CM of $\{p_i(s)\}$ such that $l(s)|c(s)$ for any other CM, $c(s)$, of $\{p_i(s)\}$.

The module of $m \times n$ matrices with entries from $Q[s]$ is denoted by $M(Q[s])$. Similarly, $M(Z[s])$ is the subset of $M(Q[s])$ when the entries are resticted to lie in $Z[s]$. $P(s) \in M(Q[s])$ is called a *polynomial matrix*. Letting $R[s]$ denote either $Z[s]$ or $Q[s]$ or, in fact, any commutative polynomial ring, $P(s) \in M(R[s])$ is said to be nonsingular if $P(s)$ is square and $\det P(s)$ is not identically zero. A nonsingular polynomial matrix, $W(s) \in M(R[s])$ is *unimodular* if its determinant is a unit of $R[s]$. In this case, therefore, $W(s)^{-1} =$

$Adj\ W(s)/\det W(s)$ is itself a polynomial matrix and therefore a unit of $M(R[s])$.

$P(s) \in M(Z[s])$ is said to be row (left) primitive, if the polynomial entries in each row are relatively primitive. For any $A(s) \in M(Z[s])$, there exists a diagonal matrix $L_A \in M(Z)$ and a row primitive matrix $P_A(s) \in M(Z[s])$ such that $A(s) = L_A\ P_A(s)$. This is known as a left content-primitive factorization of $A(s)$. The diagonal elements of L_A are called the *row contents* of the respective rows of $A(s)$. Similar statements can be made about column (right) primitive matrices.

Polynomial vectors are *linearly dependent* if they can be made proportional to each other with scalars from the field of rational fractions $a(s)/b(s)$ (i.e., $p_1(s)$ and $p_2(s)$ are linearly dependent if $p_1(s) - a(s)/b(s)\ p_2(s) = 0$). This can be simplified by writing $b(s)\ p_1(s) - a(s)\ p_2(s) = 0$ leading to the fact that polynomial vectors are dependent over the field of rational functions if they can be made dependent using only polynomial coefficients. With this in mind, the elementary row and column operations for polynomial matrices over $Q[s]$ will have the form.

1. multiplication of a row (column) by a unit of $Q[s]$, i.e., any nonzero rational.
2. add to any row (column) a polynomial multiple of any other row (column) where the polynomial $p(s) \in Q[s]$
3. interchange any two rows (columns)

Performing any of the above operations on an identity matrix results in an elementary matrix $E(s)$ over $Q[s]$. Clearly, each such $E(s)$ is unimodular and its inverse is also an elementary matrix. In fact, every unimodular matrix is a product of elementary matrices. Two polynomial matrices $A(s)$ and $B(s)$ are said to be *row equivalent* if each can be obtained from the other using a finite sequence of elementary row operations. In other words, they are related by left multiplication by a unimodular matrix $U(s)$ such that $B(s) = U(s)\ A(s)$ or $A(s) = U^{-1}(s)\ B(s)$.

Any $m \times n$ polynomial matrix $A(s) \in M(Q[s])$ of rank $r \geq 1$ is row equivalent to an upper triangular (or trapezoidal, if $m \neq n$) form. Therefore, it can be reduced by a sequence of elementary row operations to an upper triangular (trapezoidal) matrix $T(s) \in M(Q[s])$. Said another way, there exists a unimodular matrix $U(s)$ such that $U(s)\ A(s) = T(s)$ with $T(s) \in M(Q[s])$ upper triangular. If $T(s) \in M(Q[s])$ satisfies the

following conditions, it is said to be in its unique *column Hermite form.*

1. If $m > r$ then the last $m - r$ rows are identically zero
2. Each diagonal entry is monic
3. Each diagonal entry has degree greater than the entries above it.

The *pseudo-column Hermite form* for $A(s) \in M(Q[s])$ can be defined in terms of its Hermite form, $H_A(s)$, by multiplying each row of $H_A(s)$ with the smallest positive integer such that the matrix $H'_A(s)$ so obtained satisfies $H'_A(s) \in M(Z[s])$. Conversely, $H_A(s)$ can be obtained from $H'_A(s)$ by dividing each row of $H'_A(s)$ by the leading coefficient of the polynomial on the diagonal of the respective row. If the row is identically zero, do nothing. The notion of pseudo-column Hermite form gives a canonic triangular form in $M(Z[s])$ for each matrix in $M(Q[s])$.

Triangularizing Polynomial Matrices

The upper triangularization of matrices with entries from a field using a sequence of *non-singular* elementary row operations plays a key role in the theory of vector spaces. Likewise, the upper triangularization of matrices with entries from a ring using a sequence of *unimodular* elementary row operations plays a key role in the theory of matrix modules.

The transformation to an upper triangular (trapezoidal) form using elementary operations can be performed on any matrix with entries from a Euclidean ring of which $M(Q[s])$ is one example. The key feature that Euclidean rings enjoy is the Euclidean division property. Given two polynomials $a(s), b(s) \in Q[s]$, where $\deg b(s) \geq \deg a(s)$, there exist two polynomials, the quotient, $q(s)$ and the remainder, $r(s)$, such that $b(s) = q(s)\ a(s) + r(s)$ where $r(s) \equiv 0$ or $\deg r(s) < \deg a(s)$. The fact that the inequality on the degrees is strict allows one to introduce a zero into a polynomial matrix using elementary operations. As an example, consider an attempt to introduce a zero into the $(2,1)$ position of a 2×2 polynomial matrix. Suppose one has found $q(s)$ and $r(s)$ such that $b(s) = q(s)\ a(s) + r(s)$ and then computes,

$$\begin{pmatrix} 1 & 0 \\ -q(s) & 1 \end{pmatrix} \begin{pmatrix} a(s) & c(s) \\ b(s) & d(s) \end{pmatrix} = \begin{pmatrix} a(s) & c(s) \\ r(s) & d(s) - q(s)c(s) \end{pmatrix}$$

using an obviously unimodular pre-multiplication. By interchanging the rows, the same procedure can now be repeated on the resulting matrix and iterated, each step reducing the degree of the $(2,1)$ entry in view of the strict degree inequality. Clearly, the process cannot continue indefinitely without yielding a constant. If the constant is not zero, one final row exchange and the proper elementary row operation will introduce a zero into the $(2,1)$ position. It is not difficult to see that resulting polynomial in the $(1,1)$ position will be the GCD of the original polynomials $a(s)$ and $b(s)$. This process is called *Euclidean elimination* by analogy with Gaussian elimination.

Integer vs Rational Arithmetic

The above example also serves to illustrate that rational arithmetic is costly. For instance, to calculate the coefficients of polynomials of the form

$$d(s) - q(s)\ c(s) \qquad c, d, q \in Q[s]$$

one encounters the generic computation $\alpha + \beta\ \gamma \quad \alpha, \beta, \gamma \in Q$. If these are expressed as ratios of integers $\alpha = \frac{N^\alpha}{D^\alpha}$, $\beta = \frac{N^\beta}{D^\beta}$, $\gamma = \frac{N^\gamma}{D^\gamma}$, all reduced to lowest terms, then

$$\alpha + \gamma\ \delta = \frac{N^\alpha\ D^\beta\ D^\gamma + N^\beta\ N^\gamma\ D^\alpha}{D^\alpha\ D^\beta\ D^\gamma}$$

This computation requires six integer multiplications, one integer addition and the calculation of a GCD. Although there are more efficient methods (see *Knuth V2 pg. 313*), it remains a fact that rational arithmetic is computationally expensive, due in large part to the need for GCD calculations. On the other hand, if it can be arranged so that α, β and γ are all integers, then the same computation obviously requires only two integer multiplications, one integer addition and no GCD calculation. Clearly, by multiplying each row of any $A(s) \in M(Q[s])$ by a large enough integer, the denominators of every coefficient of every entry of $A(s)$ can be cancelled and such a diagonal operation is certainly unimodular in $M(Q[s])$. Because this involves a fixed overhead, assume, for convenience, that $A(s) \in M(Z[s])$. This still creates difficulties because $Z[s]$ is not a Euclidean ring. For instance, it is easy to see that the remainder of two polynomials $\in Q[s]$ with integer

coefficients has, in general, rational coefficients; consider the remainder of $2s$ after division by $3s - 1$. In other words, Euclidean division is not defined for $Z[s]$. However, $Z[s]$ is an instance of a unique factorization domain (UFD) and there exists a procedure called Pseudo-division which is defined for polynomials with coefficients in a UFD and which avoids rational arithmetic altogether.

Pseudo-division Lemma

Given polynomials $a(s), b(s) \in Z[s]$ with $\deg b(s) \geq \deg a(s)$, there exist polynomials $r(s), N(s) \in Z[s]$ and a positive integer D such that

$$D\ b(s) = N(s)\ a(s) + r(s)$$

with either $r(s) \equiv 0$ or $\deg r(s) < \deg a(s)$. Furthermore, D is the smallest such integer, i.e., for any other triple $(r'(s), N'(s), D')$ satisfying the above, it follows that $D \leq D'$. It also follows that $D \leq l_a^k$ where $k = \deg b(s) - \deg a(s) + 1$ with l_a denoting the leading coefficient of a.

Proof

In division of polynomials $b(s)$ and $a(s)$ over Q, explicit division of l_a is performed k times. Thus, if $b(s)$ and $a(s)$ start off with integer coefficients, then the only denominators which appear in the coefficients of the quotient $q(s)$ and remainder $r(s)$ are divisors of l_a^k. This suggests that it is possible to find polynomials $q(s),\ r(s) \in Z[s]$ such that $l_a^k\ b(s) = q(s)\ a(s) + r(s)$ (see *Knuth V2 pg. 407*). Denote l_a^k by D' and $q(s)$ by $N'(s)$. Write

$$r'(s) = D'\ b(s) - N'(s)\ a(s)\ \equiv\ g'\ r(s) = g'(D\ b(s) - N(s)\ a(s))$$

where $g' = GCD\{D', N'_o, \ldots, N'_{n-m}\}$. Clearly $(r(s), N(s), D) \in Z[s]$ and $D \leq D'$ •

Algorithm D - Pseudo-division of Polynomials

Given two polynomials $b(s) = b_0 s^n + b_1 s^{n-1} + \ldots + b_n,\ a(s) = a_0 s^m + a_1 s^{m-1} + \ldots + a_m \in Z[s]$ with $n \geq m$ and $a(s) \neq 0$, this algorithm computes the *smallest* pseudo-remainder

D and the pseudo-quotient $N(S)$ defined above. This algorithm computes D and $N(s)$ *directly* instead of computing D' and $N'(s)$ and then finding g'. Direct calculation of D and $N(s)$ involves computing GCD's *on the fly* which involve smaller numbers than those used to compute g'. Bigger numbers cost more in GCD calculations and given the size of the integers encountered in polynomial matrix computations, e.g., easily greater than 1000 digits, this algorithm can save a substantial amount of time.

$mnm \leftarrow \min(m, n-m)$

$g \leftarrow GCD(b_0, a_0)$

$d \leftarrow a_0/g$

$D \leftarrow d$

$b_0 \leftarrow b_0/g$

For $i = 1$ thru $n-m$ Do

  For $j = i$ thru $n-m$ Do

    $b_j \leftarrow b_j * d$

  EndDo

  For $j = 1$ thru $\min(mnm, n-m-i+1)$ Do

    $b_{j+i-1} \leftarrow b_{j+i-1} - a_j * b_i$

  EndDo

  $g \leftarrow GCD(b_i, a_0)$

  $d \leftarrow a_0/g$

  $D \leftarrow D * d$

  $b_i \leftarrow b_i/g$

EndDo

The algorithm terminates with the first $n-m+1$ coefficients of $b(s)$ overwritten according to $\{b_0, b_1, \ldots, b_{n-m}\} \leftarrow \{N_o, N_1, \ldots, N_{n-m}\}$.

Introducton of a zero in the $(2,1)$ position of the matrix in the 2×2 example can now be performed using the Pseudo-division algorithm. This freedom from rational arithmetic

is not without it's drawbacks, however. Consider triangularizing the matrix:

$$\begin{pmatrix} 1 & s & s \\ 45s & -10s-10 & 3s^2+s+10 \\ 7-5s & 6s^2-1 & 4s^2-10 \end{pmatrix}$$

Triangularization with Pseudo-division yields:

$$\begin{pmatrix} -7577325 & 0 & 1351755s^3 - 1373940s^2 + 5102550s + 7152750 \\ 0 & -89145 & 43605s^3 - 77103s^2 + 234189s + 35190 \\ 0 & 0 & -3706425s^4 + 5202000s^3 - 18532125s^2 - 15671025s - 7152750 \end{pmatrix}$$

This illustrates the main disadvantage of triangularization over $M(Z[s])$ – the coefficient growth of the polynomials. As the number of rows and columns in the matrix increases, this coefficient growth continues unabated and begins to erode the advantage of using integer arithmetic. One approach to handle this coefficient growth is to remove the content of the current row after a zero is introduced. Factoring the above matrix into its left content-primitive form $R\ A(s)$ yields:

$$R\ A(s) = \begin{pmatrix} 765 & 0 & 0 \\ 0 & 9 & 0 \\ 0 & 0 & 65025 \end{pmatrix} \begin{pmatrix} -9905 & 0 & 1767s^3 - 1796s^2 + 6670s + 9350 \\ 0 & -9905 & 4845s^3 - 8567s^2 + 26021s + 3910 \\ 0 & 0 & -57s^4 + 80s^3 - 285s^2 - 241s - 110 \end{pmatrix}$$

The superfluous left content of this matrix can then be discarded since this is equivalent to multiplying $A(s)$ by a unimodular matrix R^{-1} thereby keeping the coeffcient size to a minimum. Note that the above polynomial matrix $A(s)$ is in pseudo-column Hermite form and that the column Hermite form of $A(s)$ is:

$$\begin{pmatrix} 1 & 0 & -\frac{1767s^3}{9905} + \frac{1796s^2}{9905} - \frac{1334s}{1981} - \frac{1870}{1981} \\ 0 & 1 & -\frac{969s^3}{1981} + \frac{8567s^2}{9905} - \frac{26021s}{9905} - \frac{782}{1981} \\ 0 & 0 & s^4 - \frac{80s^3}{57} + 5s^2 + \frac{241s}{57} + \frac{110}{57} \end{pmatrix}$$

Algorithms for Triangularizing Polynomial Matrices

Algorithm T - Column Oriented Triangularization of Polynomial Matrices

Given an $N \times N$ nonsingular matrix $A \in M(Z[s])$, this algorithm overwrites A with a triangular form obtained by a sequence of unimodular, elementary row operations. It avoids rational arithmetic by using Pseudo-Euclidean division. In addition, it inhibits coefficient growth by dividing the current row (the row affected by the preceding elementary row operation step) by its row content. This algorithm operates in a column oriented fashion by successively zeroing out the entries in each column below the diagonal. This is shown pictorially below.

$$\begin{pmatrix} x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \end{pmatrix} \rightarrow \begin{pmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \\ 0 & x & x & x & x \end{pmatrix} \cdots \rightarrow \begin{pmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & x \end{pmatrix}$$

Assume there exists a pre-defined function

$$MinimumDegreeIndex(A, k) := \arg\min\{\deg A_{1,k}, \ldots, \deg A_{N,k}\}$$

which returns the index of the row of A whose k^{th} entry is a non-zero polynomial of lowest degree among the rows $\{k, k+1, \ldots, N\}$ If $A_{k,k}(s) = A_{k+1,k}(s) = A_{N,k}(s) \equiv 0$, then it returns $-\infty$

For k = 1 thru N-1 Do

$index \leftarrow MinimumDegreeIndex(A, k)$

If $index \neq -\infty$ Then

$A_{k,.} \leftrightarrow A_{index,.}$ (exchange rows k and $index$)

For $n = k+1$ thru N Do (zero out all entries in column k below $A_{k,k}$)

EndlessLoop

$num \leftarrow pseudo - quotient(A_{n,k}, A_{k,k})$

$denom \leftarrow pseudo - remainder(A_{n,k}, A_{k,k})$

$A_{n,.} \leftarrow denom * A_{n,.} - num * A_{k,.}$

$A_{n,.} \leftarrow A_{n,.}/GCD\{content(A_{n,1}), \ldots, content(A_{n,N})\}$

If $A_{n,k} \equiv 0$ then exit EndlessLoop

$A_{n,.} \leftrightarrow A_{k,.}$

End EndlessLoop

EndDo

EndIf

End

It is emphasized that this is a 'paper' algorithm. In fact, the working code based on the above is more efficient and can handle singular and non-square matrices and the entries can initially belong to $Q[s]$. However, these considerations just complicate matters and obscure the basic operation.

Algorithm M - Minor Oriented Triangularization of Polynomial Matrices

This algorithm is similar to the one above except it performs the zeroing process in a leading principal minor oriented fashion so that the algorithm consists of $N-1$ stages where the $k \times k$ leading principal submatrix is in a triangular form by the end of the k^{th} stage. Furthermore, the algorithm employs an additional substage which reduces the degrees of the polynomial entries above the diagonal *on the fly* using Pseudo-Euclidean division. The order in which the degrees are reduced is important and is based upon notions from the integer case contained in *Kannan and Bachem, 1979.* The order is shown pictorially below. The output matrix is in its unique *Pseudo-Hermite form*, not simply triangularized, i.e., it is a triangular matrix with entries above the diagonal of degree less than the diagonal entry. It is technically not in Hermite form since the diagonal entries are not monic. But this form is easily obtained by left multiplication with the appropriate diagonal matrix of rational numbers, a unimodular matrix with respect to $Q[s]$.

$$\begin{pmatrix} x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \end{pmatrix} \rightarrow \begin{pmatrix} x & 1 & x & x & x \\ 0 & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \end{pmatrix} \rightarrow \begin{pmatrix} x & 2 & 3 & x & x \\ 0 & x & 1 & x & x \\ 0 & 0 & x & x & x \\ x & x & x & x & x \\ x & x & x & x & x \end{pmatrix}$$

$$\longrightarrow \begin{pmatrix} x & 4 & 5 & 6 & x \\ 0 & x & 2 & 3 & x \\ 0 & 0 & x & 1 & x \\ 0 & 0 & 0 & x & x \\ x & x & x & x & x \end{pmatrix} \cdots \longrightarrow \begin{pmatrix} x & x & x & x & x \\ 0 & x & x & x & x \\ 0 & 0 & x & x & x \\ 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & x \end{pmatrix}$$

For $k = 2$ thru N Do

For $n = 1$ thru $k-1$ Do (triangularize the $k \times k^{th}$ leading principal minor)

if $\deg A_{n,n} > \deg A_{k,n}$ Then $A_{k,.} \leftrightarrow A_{n,.}$

EndlessLoop

$num \leftarrow pseudoquotient(A_{k,n}, A_{n,n})$

$denom \leftarrow pseudoremainder(A_{k,n}, A_{n,n})$

$A_{k,.} \leftarrow denom * A_{k,.} - num * A_{n,.}$

$A_{k,.} \leftarrow A_{k,.}/GCD\{content(A_{k,1}), \ldots, content(A_{k,N})\}$

If $A_{k,n} \neq 0$ then $A_{k,.} \leftrightarrow A_{n,.}$ else Exit EndlessLoop

End EndlessLoop

End Do

For $i = -1$ thru $-k+1$ step -1 Do (reduce degs of abv diag polys in $k \times k^{th}$ minor)

For $j = i+1$ thru 0 Do

If $\deg A_{k+i,k+j} \geq \deg A_{k+j,k+j}$ Then

$num \leftarrow pseudoquotient(A_{k+i,k+j}, A_{kk+j,kk+j})$

$denom \leftarrow pseudoremainder(A_{k+i,k+j}, A_{kk+j,kk+j})$

$A_{k+i,.} \leftarrow denom * A_{k+i,.} - num * A_{k+j,.}$

$A_{k+i,.} \leftarrow A_{k+i,.}/GCD\{content(A_{k+i,1}), \ldots, content(A_{k+i,N})\}$

EndIf

EndDo

EndDo

EndDo

Simulation Results

MACSYMA is a Lisp based system for performing formal, symbolic calculations using both error-free and arbitrary artihmetic. Since it is Lisp based, MACSYMA runs the fastest on *Lisp machines* - computers which have which have the Lisp instructions hard-coded into their microprocessors, such as the Texas Instruments Explorer II. *Mathematica* is a new computer algebra system which has similar capabilities of MACSYMA, but is written in 'C'. We have been running *Mathematica* on the NeXT Machine, but its relatively slow microprocessor and small memory severly limit its performance.

Simulations were performed to determine the average time required to triangularize square polynomial matrices and the maximum coefficient length using both *Algorithm T* and *Algorithm M* (see attached graphs). Each matrix had polynomial entries with randomly generated 1–2 digit coefficients. Runs were parameterized by the degree of the polynomial, which ranged from 1 to 6, and the dimension of the matrix, which ranged form 2 to 16.

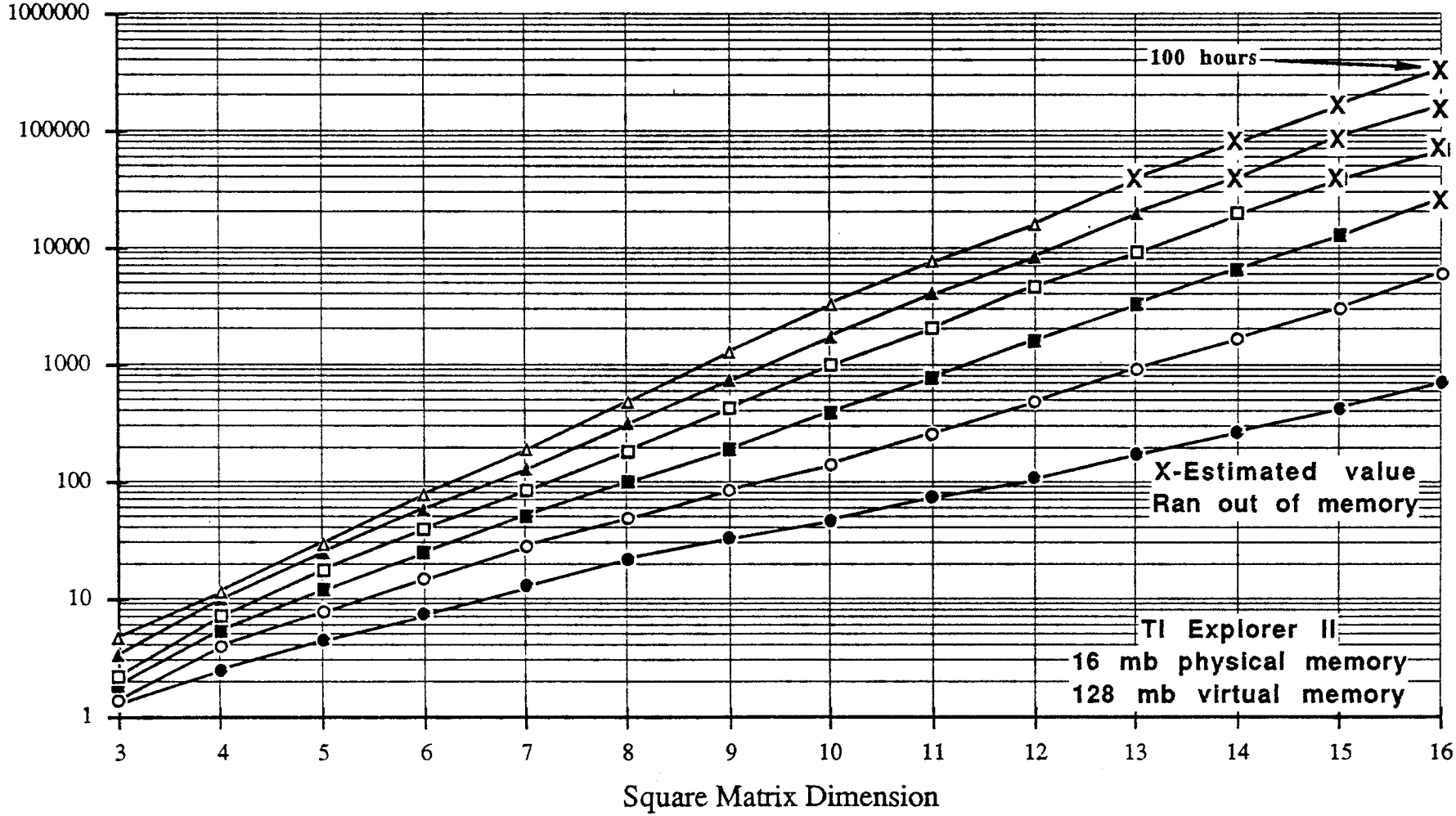
These simulations were run on a Texas Intruments Explorer II with 16 mb of physical memory and 128 mb of virtual memory using the MACSYMA version of our algorithms. The graphs represent the results of the simulations averaged over 7 runs. The results indicate that *Algorithm T* was moderately faster than *Algorithm M* in triangularizing matrices up to 9×9. At that point *Algorithm T* was still faster for triangularizing matrices with lower degree polynomials, but slower in the higher degree polynomials. This can be attributed to the fact that *Algorithm M* requires less memory during computations due to its substage which reduces the degrees of the polynomials above the diagonal on the fly. Therefore costly garbage collections, a technique of freeing dynamically allocated memory, are reduced. This is further shown by the fact that *Algorithm T* ran out of memory while attempting to triangularize a 13×13 matrix with degree 6 polynomial entries while *Algorithm M* completed a 16×16 matrix with degree 6 entries.

It appears that both of these algorithms run close to exponential time. The slopes of the semi-log plots of the timings increase slightly with increasing polynomial degree. The maximum coefficient length was approximately the same for each algorithm and the coefficient growth appears to be sub-exponential with increasing matrix dimension. A

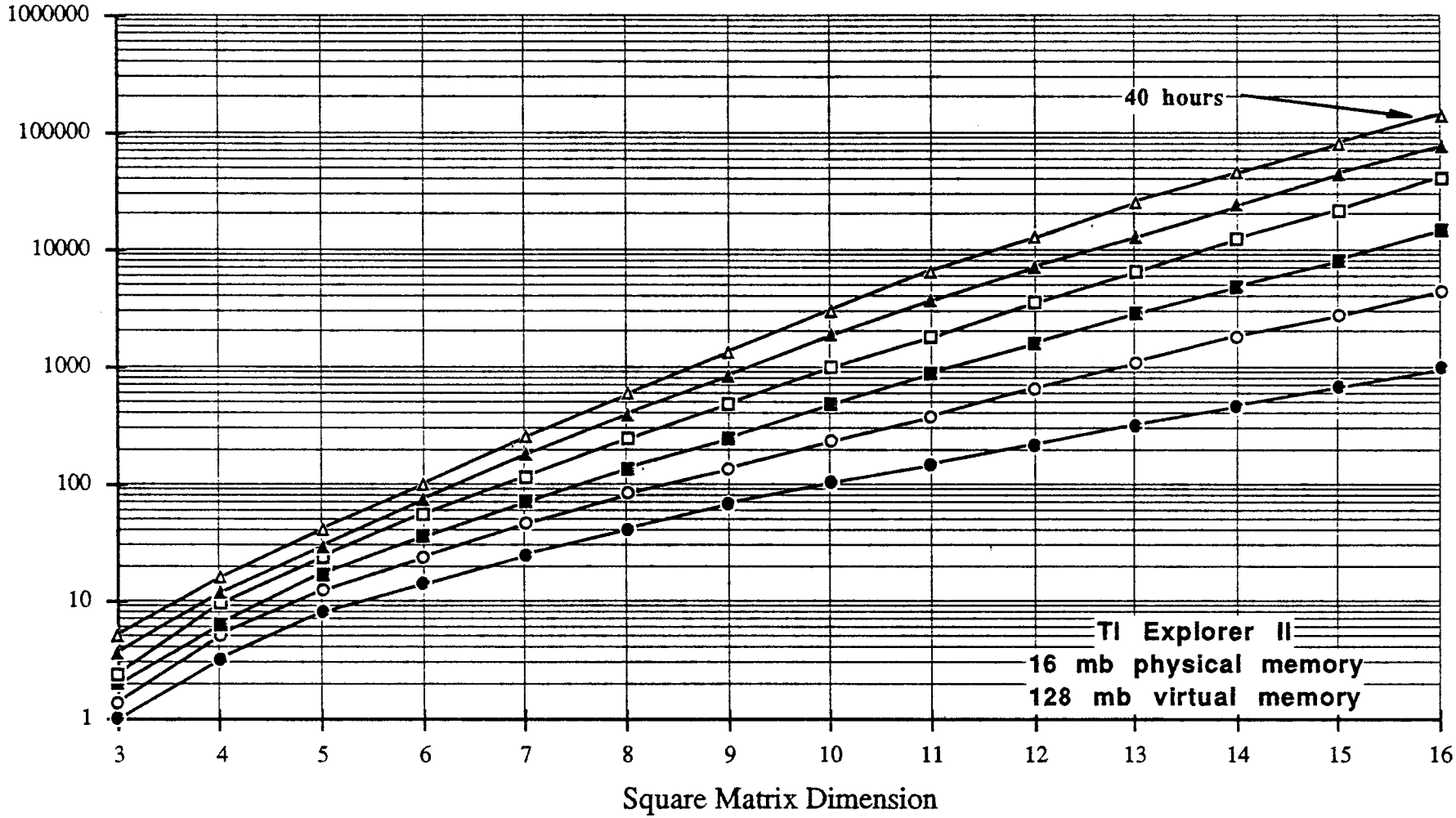
16×16 matrix with degree 6 polynomials is the largest that has been attempted with *Algorithm M*. It required 40 hours to triangularize with the resulting matrix having a maximum coefficient length of 2115 digits.

Although *Algorithm T* was faster than *Algorithm M* on the smaller matrices, it did not have the overhead of putting the matrix in a canonic form in the process. *Algorithm M* leaves the matrix in a unique Pseudo-Hermite form as described earlier. The output matrix of *Algorithm T* requires the application of an additional algorithm to reduce the degree of its above diagonal polynomials in order to put it in a Pseudo-Hermite form.

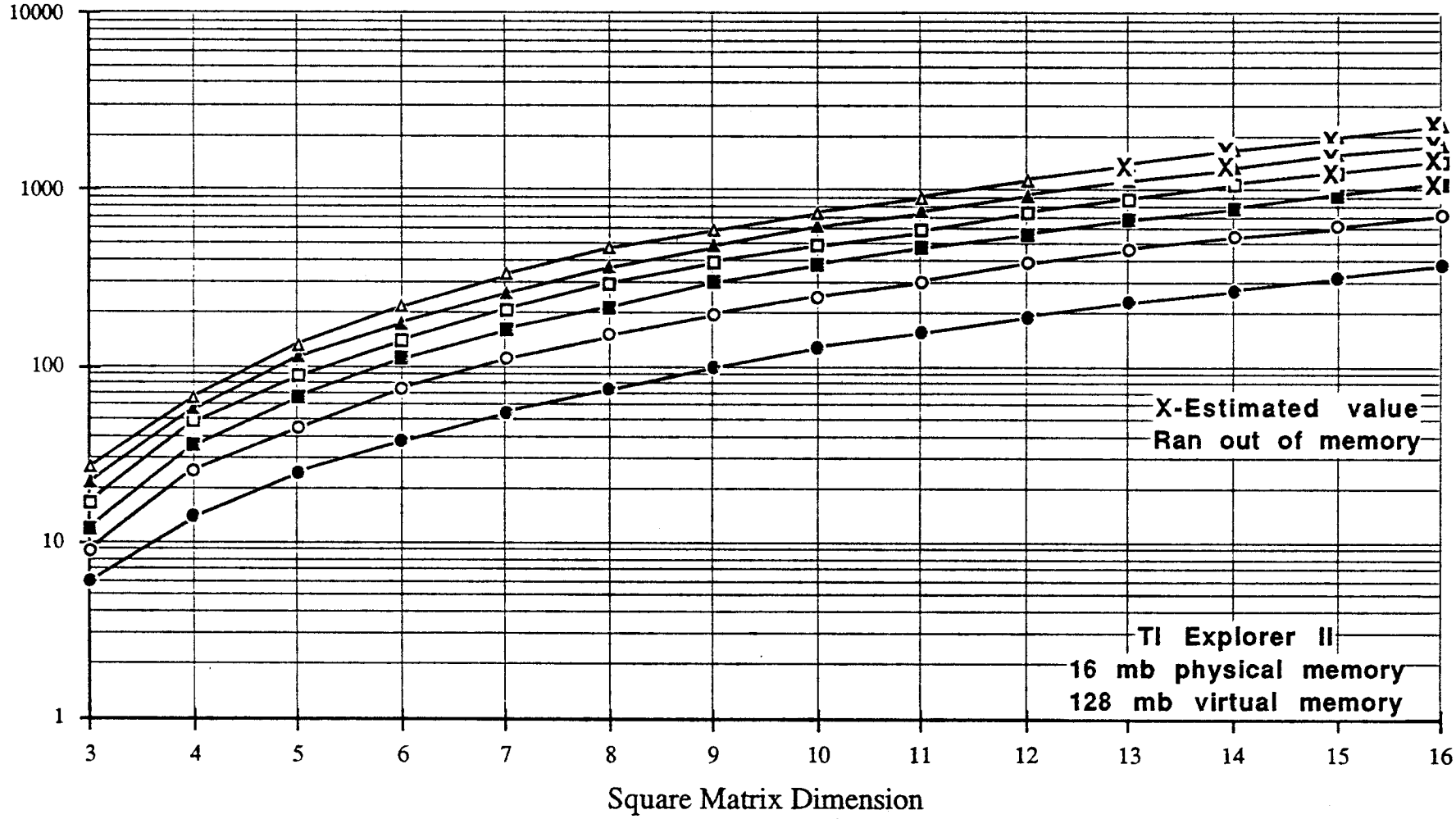
Time to Triangularize (sec) - Column Oriented Algorithm
Polynomial Degrees 1 thru 6



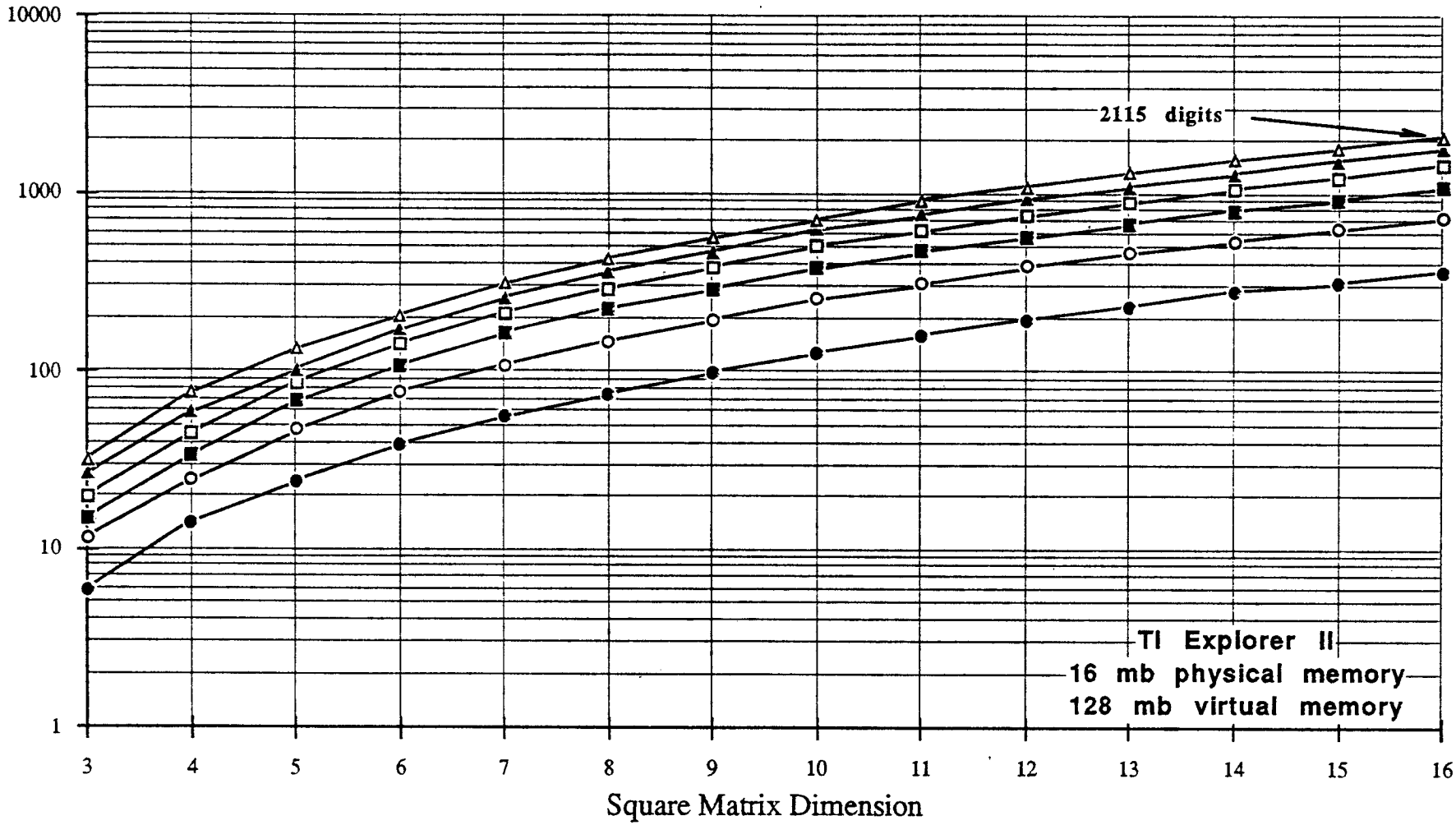
Time to Triangularize (sec) - Minor Oriented Algorithm
Polynomial Degrees 1 thru 6



Maximum Coefficient Length (# of digits) - Column Oriented Algorithm
Polynomial Degrees 1 thru 6



Maximum Coefficient Length (# of digits) - Minor Oriented Algorithm
Polynomial Degrees 1 thru 6



Summary of Functions

The following is a summary of the high-level auxiliary programs which we have to date implemented in MACSYMA and *Mathematica*. They perform most of the common, high-level tasks arising in the control system design process.

- *RightMatrixFraction*$(H(s))$—Computes a right matrix fraction description of the transfer function matrix $H(s)$, i.e., computes the matrices $N(s), D(s)$ such that $H(s) = N(s)\ D(s)^{-1}$. The LeftMatrixFraction description is analogously computed.

- *Bezout*$(N(s), D(s))$ — Finds the homogenous and particular solutions to the Bezout equation, i.e., finds polynomial matrices $Xp(s), Yp(s), Xh(s), Yh(s)$ such that $Xp(s)\ X(s) + Yp(s)\ N(s) = I$ and $Xh(s)\ X(s) + Yh(s)\ N(s) = 0$. Used for designing feedback compensators in the frequency domain.

- *ColumnReduce*$(D(s))$—Column reduces the polynomial matrix $D(s)$, i.e., multiplies $D(s)$ by an appropriate unimodular matrix such that the matrix of leading coefficients of its entries is nonsingular. RowReduce is analogously computed.

- *Controller*$(H(s))$—Finds a controller form realization of the transfer function matrix $H(s)$. Controllability, Observer and Observability realizations are analogously computed.

- *Hermite*$(N(s))$—Finds the Hermite form of the polynomial matrix $N(s)$.

- *RightCoprime*$(N(s), D(s))$—Determines the greatest common right divisor of the polynomial matrices $N(s)$ and $D(s)$. If it is not unimodular, it is *pulled out* of both matrices making them right coprime. Used for finding *minimal* realizations. LeftCoprime is analogously computed.

- *Smith*$(N(s))$—Finds the Smith form of the polynomial matrix $N(s)$. This is a canonic, diagonal form of a polynomial matrix.

- *SmithMcMillan*$(H(s))$—Finds the Smith-McMillan form of the transfer function matrix $H(s)$. This is a canonic, rational, diagonal form of a matrix whose entries are ratios of polynomials.

References and Bibliography

Bariess, E.H. "Computational Solutions of Matrix Problems over an Integral Domain" *J. Inst. Maths Applics* V10, 69-104, 1972

Bariess, E.H. "Sylvester's Identity and Multistep Integer-Preserving Gaussian Elimination" *Math. Comp.* V22, 565-578, 1968

Brown, W.S. "On Euclid's Algorithm and the Computation of Polynomial Greatest Common Divisors" *J. ACM* V18 (4) 478-504 Oct 71

Brown, W.S. & J.F. Traub "On Euclid's Algorithm and the Theory of Subresultants" *J. ACM* V18 (4) 505-514 Oct 71

Chou, T.J. & G.E. Collins "Algorithms for the Solution of Systems of Linear Diophantine Equations" *Siam. J. Comp.* V11 (4) 687-708 Nov 82

Collins, G.E. "Subresultants and Reduced Polynomial Remainder Sequences" *J. ACM* V14 (1) 128-142 Jan 67

Gantmakher, F.R. *Theory of Matrices* New York: Chelsea, 1959

Gregory, R.T., & E.V. Krishnamurthy *Methods and Applications of Error-Free Computation* Berlin: Springer, 1984

Hartley, B. & T.O. Hawkes *Rings, Modules and Linear Algebra* London: Chapman and Hall, 1970

Hungerford, T.W. *Algebra* Berlin: Springer, 1974

Kailath, T. *Linear Systems* Englewood Cliffs: Prentice-Hall, 1980

Kannan, R. "Solving Systems of Linear Equations over Polynomials" Report CMU-CS-83-165, Dept. of Comp. Sci., Carnegie-Mellon University, Pittsburgh, 1983

Kannan, R. & A. Bachem "Polynomial Algorithms for Computing the Smith and Hermite Normal Forms of an Integer Matrix" *Siam. J. Comp.* V8 (4) 499-507 Nov 79

Keng, H.L. *Introduction to Number Theory* Berlin: Springer, 1982

Knuth, D.E. *The Art of Computer Programming, Vol. 2* Reading, Mass.:Addison Wesley,1981

Krishnamurthy, E.V. *Error-Free Polynomial Matrix Computations* Berlin: Springer, 1985

Lipson, J.D. *Elements of Algebra and Algebraic Computing* Reading: Addison-Wesley, 1981

MacDuffee, C.C. *The Theory of Matrices* New York: Chelsea, 1950

McClellan, M.T. "The Exact Solution of Systems of Linear Equations with Polynomial Coefficients" *J. ACM* V20 (4) 563-588 Oct 73

Newman, M. *Integral Matrices* New York: Academic Press, 1972

Vidyasagar, M.*Control System Synthesis* Cambridge: MIT Press, 1985

Wolovich, W.A.*Linear Multivariable Systems* Berlin: Springer, 1974