Evaluating parametric holonomic sequences using rectangular splitting

Fredrik Johansson

ISSAC 2014, Kobe, Japan

Goal

Compute the entry $c(n)$ in a sequence satisfying a linear recurrence equation

$$\underbrace{c(k+1)}_{\text{vector}} = \underbrace{M(k)}_{\substack{\text{square matrix} \\ \text{entries polynomials in } k}} \cdot \underbrace{c(k)}_{\text{vector}}$$

Goal

Compute the entry $c(n)$ in a sequence satisfying a linear recurrence equation

$$\underbrace{c(k+1)}_{\text{vector}} = \underbrace{M(k)}_{\substack{\text{square matrix} \\ \text{entries polynomials in } k}} \cdot \underbrace{c(k)}_{\text{vector}}$$

Examples

$$n!, \qquad \prod_{k=0}^{n-1}(x+k), \qquad \sum_{k=1}^{n} \frac{1}{x+k},$$

$$\exp(x) \approx \sum_{k=0}^{n} \frac{x^k}{k!}, \qquad J_n(x) \approx \sum_{k=0}^{n} \frac{(-1)^k}{k!(k+n)!}\left(\frac{x}{2}\right)^{2k+n}$$

How to do it

Naively

How to do it

Naively

$$c(1) = M(0)c(0)$$

How to do it

Naively

$$c(1) = M(0)c(0)$$
$$c(2) = M(1)c(1)$$

How to do it

Naively

$$c(1) = M(0)c(0)$$
$$c(2) = M(1)c(1)$$
$$c(3) = M(2)c(2)$$
$$\vdots$$

How to do it

Naively

$$c(1) = M(0)c(0)$$
$$c(2) = M(1)c(1)$$
$$c(3) = M(2)c(2)$$
$$\vdots$$

Cleverly

Compute

$$M(n-1)M(n-2)\cdots M(1)M(0)$$

Then multiply by the initial vector $c(0)$

Binary splitting

Example: $M(k) = (x + k), \qquad n = 4$

Binary splitting

Example: $M(k) = (x + k), \qquad n = 4$

$$(x+3) \quad (x+2) \qquad (x+1) \quad (x+0)$$
$$\searrow \quad \swarrow \qquad \searrow \quad \swarrow$$

Binary splitting

Example: $M(k) = (x + k)$, $\qquad n = 4$

$$(x+3) \quad (x+2) \qquad (x+1) \quad (x+0)$$

$$\searrow \quad \swarrow \qquad\qquad \searrow \quad \swarrow$$

$$(x^2 + 5x + 6) \qquad\qquad (x^2 + x)$$

$$\searrow \qquad\qquad\qquad \swarrow$$

Binary splitting

Example: $M(k) = (x+k), \qquad n = 4$

$$\begin{array}{ccccccc}
(x+3) & & (x+2) & & (x+1) & & (x+0) \\
& \searrow & \swarrow & & & \searrow \ \swarrow & \\
& (x^2+5x+6) & & & & (x^2+x) & \\
& & \searrow & & \swarrow & & \\
& & & (x^4+6x^3+11x^2+6x) & & &
\end{array}$$

Binary splitting

Example: $M(k) = (x+k), \qquad n = 4$

$$
\begin{array}{ccccccc}
(x+3) & & (x+2) & & (x+1) & & (x+0) \\
& \searrow & \swarrow & & & \searrow \; \swarrow & \\
& (x^2+5x+6) & & & & (x^2+x) & \\
& & \searrow & & \swarrow & & \\
& & & (x^4+6x^3+11x^2+6x) & & &
\end{array}
$$

Useful if **the cost grows with the entries**

$R[x]$	$O(\mathrm{M}(n)\log n)$	$= O\tilde{}(n)$ R-operations
$\mathbb{Z}$	$O(\mathrm{M}(n\log n)\log n)$	$= O\tilde{}(n)$ bit operations

Fast multipoint evaluation

Example: $M(k) = (k+1), \qquad n = 9$

Fast multipoint evaluation

Example: $M(k) = (k+1), \qquad n = 9$

$$P = (k+3)(k+2)(k+1) = k^3 + 6k^2 + 11k + 6$$

binary splitting

Fast multipoint evaluation

Example: $M(k) = (k+1), \qquad n = 9$

$$P = (k+3)(k+2)(k+1) = k^3 + 6k^2 + 11k + 6$$

binary splitting

$$[P(6), P(3), P(0)] = [504, 120, 6]$$

fast multipoint evaluation

Fast multipoint evaluation

Example: $M(k) = (k+1), \qquad n = 9$

$$P = (k+3)(k+2)(k+1) = k^3 + 6k^2 + 11k + 6$$

binary splitting

$$[P(6), P(3), P(0)] = [504, 120, 6]$$

fast multipoint evaluation

$P(6)P(3)P(0) = 362880$ repeated multiplication

Fast multipoint evaluation

Example: $M(k) = (k+1), \qquad n = 9$

$P = (k+3)(k+2)(k+1) = k^3 + 6k^2 + 11k + 6$
binary splitting

$[P(6), P(3), P(0)] = [504, 120, 6]$
fast multipoint evaluation

$P(6)P(3)P(0) = 362880$ repeated multiplication

Useful if **arithmetic operations have fixed cost**:
$O(\mathrm{M}(n^{1/2}) \log n) = O\tilde{~}(n^{1/2})$ operations
(Bostan, Gaudry, Schost, 2007): $O(\mathrm{M}(n^{1/2}))$

Parametric sequences

$M(k) = M(x, k)$, where entries of M are in $R[x][k]$

Evaluate at some given **"expensive"** value of the **parameter** x

Parametric sequences

$M(k) = M(x, k)$, where entries of M are in $R[x][k]$

Evaluate at some given **"expensive"** value of the **parameter** x

$c = 42 \in R$
$x = 3.14159265358979323846264338327950288 4$

Parametric sequences

$M(k) = M(x, k)$, where entries of M are in $R[x][k]$

Evaluate at some given **"expensive"** value of the **parameter** x

$c = 42 \in R$
$x = 3.14159265358979323846264338327950288 4$

Distinguish between operations

Coefficient	GOOD	$c + c, \quad c \cdot c$
Scalar	OK	$x + x, \quad c \cdot x$
Nonscalar	BAD	$x \cdot x$

Rectangular splitting

Evaluate the polynomial $\sum_{i=0}^{n} \square x^i$

Rectangular splitting

Evaluate the polynomial $\sum_{i=0}^{n} \square x^i$

$$
\begin{array}{ccccccccccc}
(& \square & + & \square x & + & \square x^2 & + & \square x^3 &) & & + \\
(& \square & + & \square x & + & \square x^2 & + & \square x^3 &) & x^4 & + \\
(& \square & + & \square x & + & \square x^2 & + & \square x^3 &) & x^8 & + \\
(& \square & + & \square x & + & \square x^2 & + & \square x^3 &) & x^{12} &
\end{array}
$$

Rectangular splitting

Evaluate the polynomial $\sum_{i=0}^{n} \square x^i$

$$
\begin{array}{l}
(\ \square + \square x + \square x^2 + \square x^3\ ) \qquad + \\
(\ \square + \square x + \square x^2 + \square x^3\ )\ x^4 + \\
(\ \square + \square x + \square x^2 + \square x^3\ )\ x^8 + \\
(\ \square + \square x + \square x^2 + \square x^3\ )\ x^{12}
\end{array}
$$

Rectangular splitting

Evaluate the polynomial $\sum_{i=0}^{n} \square x^i$

$$
\begin{array}{l}
(\square + \square x + \square x^2 + \square x^3) \quad\quad\;\; + \\
(\square + \square x + \square x^2 + \square x^3)\, x^4 + \\
(\square + \square x + \square x^2 + \square x^3)\, x^8 + \\
(\square + \square x + \square x^2 + \square x^3)\, x^{12}
\end{array}
$$

- $O(n)$ scalar operations, and
- $O(n^{1/2})$ nonscalar multiplications

(Paterson-Stockmeyer, 1973)

Rectangular splitting for sequences

Expand $P = M(x, n-1) \cdots M(x,1) M(x,0)$

using binary splitting

Evaluate each entry in P

using rectangular splitting

Rectangular splitting for sequences

Expand $P = M(x, n-1) \cdots M(x, 1) M(x, 0)$

using binary splitting

Evaluate each entry in P

using rectangular splitting

- $O(\mathsf{M}(n) \log n)$ coefficient operations
- $O(n)$ scalar operations
- $O(n^{1/2})$ nonscalar operations

This is actually bad

$$M = (x + k), \quad n = 16$$

$$P = x^{16} + 120x^{15} + 6580x^{14} + 218400x^{13} + 4899622x^{12} + 78558480x^{11} + 928095740x^{10} + 8207628000x^{9} + 54631129553x^{8} + 272803210680x^{7} + 1009672107080x^{6} + 2706813345600x^{5} + 5056995703824x^{4} + 6165817614720x^{3} + 4339163001600x^{2} + 1307674368000x$$

Coefficients grow to $O(n \log n)$ bits.
Scalar multiplications can become **slower** than nonscalar multiplications!

Improved rectangular splitting

Expand $O(n^{1/2})$ polynomials of degree $O(n^{1/2})$

Improved rectangular splitting

Expand $O(n^{1/2})$ polynomials of degree $O(n^{1/2})$

$$\begin{array}{lllll}
(M(x,15) & M(x,14) & M(x,13) & M(x,12)) & \cdot \\
(M(x,11) & M(x,10) & M(x,9) & M(x,8)) & \cdot \\
(M(x,7) & M(x,6) & M(x,5) & M(x,4)) & \cdot \\
(M(x,3) & M(x,2) & M(x,1) & M(x,0)) &
\end{array}$$

Improved rectangular splitting

Expand $O(n^{1/2})$ polynomials of degree $O(n^{1/2})$

$$
\begin{aligned}
&(x^4 + 54x^3 + 1091x^2 + 9774x + 32760)\cdot\\
&(x^4 + 38x^3 + 539x^2 + 3382x + 7920)\cdot\\
&(x^4 + 22x^3 + 179x^2 + 638x + 840)\cdot\\
&(x^4 + 6x^3 + 11x^2 + 6x + 0)
\end{aligned}
$$

Improved rectangular splitting

Expand $O(n^{1/2})$ polynomials of degree $O(n^{1/2})$

$$
\begin{aligned}
&(x^4 + 54\, x^3 + 1091\, x^2 + 9774\, x + 32760) \cdot \\
&(x^4 + 38\, x^3 + 539\, x^2 + 3382\, x + 7920) \cdot \\
&(x^4 + 22\, x^3 + 179\, x^2 + 638\, x + 840) \cdot \\
&(x^4 + 6\, x^3 + 11\, x^2 + 6\, x + 0)
\end{aligned}
$$

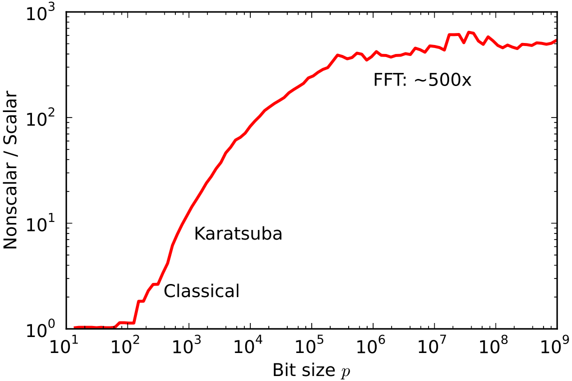
Improved rectangular splitting

Expand $O(n^{1/2})$ polynomials of degree $O(n^{1/2})$

$$
\begin{aligned}
&(x^4 + 54x^3 + 1091x^2 + 9774x + 32760)\,\cdot \\
&(x^4 + 38x^3 + 539x^2 + 3382x + 7920)\,\cdot \\
&(x^4 + 22x^3 + 179x^2 + 638x + 840)\,\cdot \\
&(x^4 + 6x^3 + 11x^2 + 6x + 0)
\end{aligned}
$$

- $O(\mathrm{M}(n)\log n)$ coefficient operations
- $O(n)$ scalar operations
- $O(n^{1/2})$ nonscalar operations
- $O(n^{1/2}\log n)$-bit coefficients

Numerical evaluation



Asymptotic speedup when the parameter x is a real number with $p \sim n$ bits: $500^{1/2} \approx 20$ (no further improvement when step length exceeds $\sim 500^{1/2}$)

Hypergeometric series summation

Smith (1989): rectangular splitting with content removal $\to$ $O(\log n)$ bit coefficients

$$\begin{aligned}\exp(x) \approx \quad & \left(1 + \tfrac{1}{1}\left(x + \tfrac{1}{2}\left(x^2 + \tfrac{1}{3}x^3\right)\right)\right) \\ + \tfrac{x^4}{4!} & \left(1 + \tfrac{1}{5}\left(x + \tfrac{1}{6}\left(x^2 + \tfrac{1}{7}x^3\right)\right)\right) \\ + \tfrac{x^8}{8!} & \left(1 + \tfrac{1}{9}\left(x + \tfrac{1}{10}\left(x^2 + \tfrac{1}{11}x^3\right)\right)\right)\end{aligned}$$

Hypergeometric series summation

Smith (1989): rectangular splitting with content removal $\rightarrow O(\log n)$ bit coefficients

$$\begin{aligned}
\exp(x) \approx \quad & \left(1 + \tfrac{1}{1}\left(x + \tfrac{1}{2}\left(x^2 + \tfrac{1}{3}x^3\right)\right)\right) \\
+\tfrac{x^4}{4!} \quad & \left(1 + \tfrac{1}{5}\left(x + \tfrac{1}{6}\left(x^2 + \tfrac{1}{7}x^3\right)\right)\right) \\
+\tfrac{x^8}{8!} \quad & \left(1 + \tfrac{1}{9}\left(x + \tfrac{1}{10}\left(x^2 + \tfrac{1}{11}x^3\right)\right)\right)
\end{aligned}$$

Our algorithm: gives larger coefficients, but works for more general sequences, and avoids divisions

Smith's algorithm for rising factorials

Smith (2001): use
$\Delta = \prod_{i=0}^{3}(x + k + 4 + i) - \prod_{i=0}^{3}(x + k + i)$

$$\begin{aligned}\Delta = &(840 + 632k + 168k^2 + 16k^3)\\ &+ (632 + 336k + 48k^2)x\\ &+ (168 + 48k)x^2\\ &+ 16x^3\end{aligned}$$

Smith's algorithm for rising factorials

Smith (2001): use
$\Delta = \prod_{i=0}^{3}(x+k+4+i) - \prod_{i=0}^{3}(x+k+i)$

$$\begin{aligned}\Delta = {} & (840 + 632k + 168k^2 + 16k^3) \\ & + (632 + 336k + 48k^2)x \\ & + (168 + 48k)x^2 \\ & + 16x^3\end{aligned}$$

Our algorithm: generalizes this to arbitrary step length (plus complexity analysis), arbitrary holonomic sequences; slight simplification

Rising factorial benchmark

Compute $\prod_{k=0}^{n-1}(x+k)$ where x is a real number with p bits of precision

Rising factorial benchmark

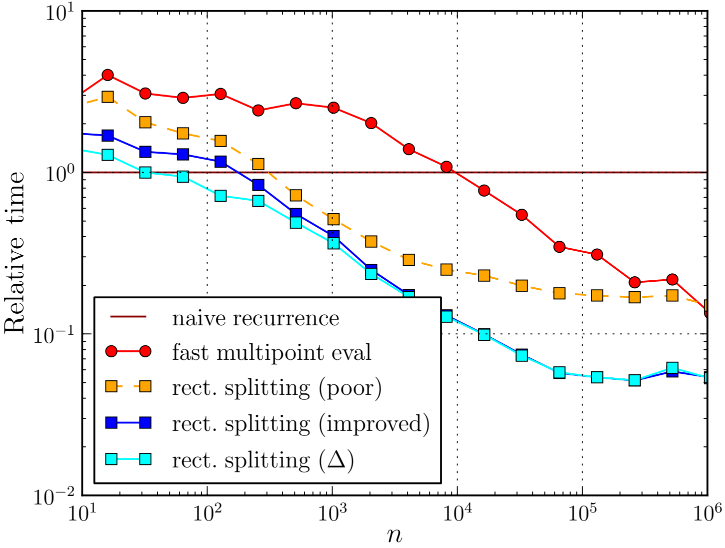
Compute $\prod_{k=0}^{n-1}(x+k)$ where x is a real number with p bits of precision

- Naive algorithm
- Algorithm 1: fast multipoint evaluation
- Algorithm 2: poor rectangular splitting
- Algorithm 3: improved rectangular splitting
- Algorithm 4: difference version

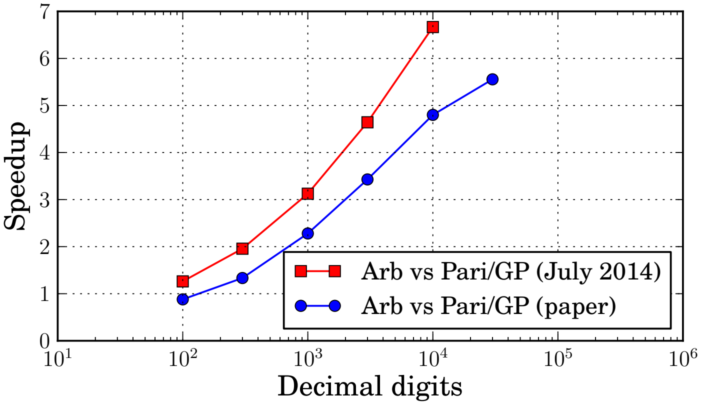
Algorithm 3 and 4 use step length $\min(0.2p^{0.4}, n^{0.5})$

In the benchmark: $p = 4n$

Rising factorial benchmark results



Speedup for $\Gamma(x)$



$\Gamma(x) = \Gamma(x+n)/(x(x+1)\cdots(x+n-1))$, Stirling series for large $x+n$

Asymptotically fast gamma function

$$\Gamma(x) \approx \gamma(x, N) = x^{-1} N^x e^{-N} {}_1F_1(1, 1 + x, N)$$

Asymptotically fast gamma function

$$\Gamma(x) \approx \gamma(x, N) = x^{-1} N^x e^{-N} \, {}_1F_1(1, 1+x, N)$$

Partial sums for ${}_1F_1(1, 1+x, N)$ satisfy order-2 recurrence with

$$M(x,k) = \frac{1}{1+k+x} \begin{pmatrix} 1+k+x & 1+k+x \\ 0 & N \end{pmatrix}$$

Asymptotically fast gamma function

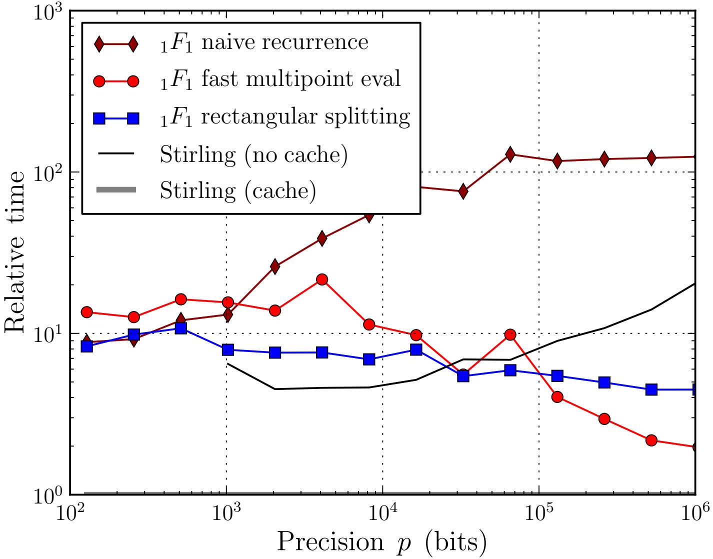
$$\Gamma(x) \approx \gamma(x, N) = x^{-1} N^x e^{-N} \, {}_1F_1(1, 1 + x, N)$$

Partial sums for ${}_1F_1(1, 1 + x, N)$ satisfy order-2 recurrence with

$$M(x, k) = \frac{1}{1 + k + x} \begin{pmatrix} 1 + k + x & 1 + k + x \\ 0 & N \end{pmatrix}$$

For p-bit precision: $n \sim N \sim p$, $\tilde{O}(p^{1.5})$ with fast multipoint evaluation

Comparison of gamma function algorithms



Summary

Efficient rectangular splitting for parametric sequences:

Summary

Efficient rectangular splitting for parametric sequences:

- Simple, works for a very general class of sequences

Summary

Efficient rectangular splitting for parametric sequences:

- Simple, works for a very general class of sequences
- Asymptotically slower than fast multipoint evaluation, but competitive in practice

Summary

Efficient rectangular splitting for parametric sequences:

- Simple, works for a very general class of sequences
- Asymptotically slower than fast multipoint evaluation, but competitive in practice
- Generalizes two different algorithms by Smith

Summary

Efficient rectangular splitting for parametric sequences:

- Simple, works for a very general class of sequences
- Asymptotically slower than fast multipoint evaluation, but competitive in practice
- Generalizes two different algorithms by Smith
- Fastest available high-precision implementation of the gamma function