New Tricks for an Old Dog: Archimedes, Gauss, and the Fast Computation of Stochastic Inverse Transcendentals

Kevin Long and Kaleb D. McKale

Department of Mathematics and Statistics
Texas Tech University

DOE Applied Math Meeting
Oct 2011
Problem: compute inverse transcendental functions of polynomial chaos expansion (PCE)

<table>
<thead>
<tr>
<th>Most classical methods for computing transcendental functions are not useable with PCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Computation of double-precision transcendental functions usually involves piecewise functions</td>
</tr>
<tr>
<td>- Example: computation of arctangent</td>
</tr>
<tr>
<td>- $0 &lt; x \leq 1$: do something fast and accurate (e.g., Padé approximant, Chebyshev fit)</td>
</tr>
<tr>
<td>- $x &gt; 1$: map into $[0, 1]$ using $\arctan(x^{-1}) = \frac{\pi}{2} - \arctan x$</td>
</tr>
<tr>
<td>- $x &lt; 0$: use odd symmetry to map to $[0, \infty)$</td>
</tr>
<tr>
<td>- Piecewise approximations not suitable in PCE work</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Workarounds</th>
</tr>
</thead>
<tbody>
<tr>
<td>- NISP</td>
</tr>
<tr>
<td>- Line integration (reduction to 1D quadrature or ODE)</td>
</tr>
</tbody>
</table>
Why are stochastic inverse transcendentals challenging?

- Taylor series approximations not convergent
  - Support of PDF can extend beyond radius of convergence
- Non-polynomial integrands make line integration expensive
  - Line integration requires linear solve (or worse) at each quadrature point
  - Convergence of quadrature can be slow
- Non-polynomial behavior requires high order PCE with accurate coefficients
  - $\tan^{-1}$ is bounded
  - log is singular at zero
  - $\sin^{-1}$ non-differentiable at $\pm 1$
Line integration method (Debusschere et al.) reduces computation of stochastic transcendentals to 1D quadrature

- **Arctangent**
  \[ \tan^{-1} u(\xi) = u(\xi) \int_0^1 \frac{dt}{1 + u(\xi)^2 t^2} \]

- **Arcsine**
  \[ \sin^{-1} u(\xi) = u(\xi) \int_0^1 \frac{dt}{\sqrt{1 - u(\xi)^2 t^2}} \]

- **Logarithm**
  \[ \log u(\xi) = (u(\xi) - 1) \int_0^1 \frac{dt}{1 + (u(\xi) - 1)t} \]
A different approach to computing inverse transcendental: Borchardt-Gauss (BG) iterated means

Example: BG computation of arctangent

Initialize:

\[ a_0 = 1 \quad g_0 = (1 + x^2)^{1/2} \]

Loop: syncopated arithmetic-geometric mean

\[ a_{n+1} = \frac{1}{2} (a_n + g_n), \quad g_{n+1} = \sqrt{g_n a_{n+1}} \]

\[ B(a_0, g_0) = \lim_{n \to \infty} g_n \]

Postprocess:

\[ \arctan x = \frac{x}{B(a_0, g_0)} \]

- Requires only addition/subtraction, multiplication/division, square root
- This is a very old idea: Archimedes developed a similar algorithm using harmonic means instead of geometric means.
Why it works: half-angle identities

**BG computation of arctangent**

- If $\theta = \arctan(x)$ then $\sin(\theta) = \frac{x}{\sqrt{1+x^2}}$, $\cos(\theta) = \frac{1}{\sqrt{1+x^2}}$
- $\theta = \lim_{n \to \infty} 2^n \sin \left( \frac{\theta}{2^n} \right)$
- Iterated half-angle identity:

\[ 2^n \sin \left( \frac{\theta}{2^n} \right) = \frac{\sin(\theta)}{\prod_{k=1}^{n} \cos \left( \frac{\theta}{2^k} \right)} \]

- Sequence of geometric means $g_n$ goes to

\[ \prod_{k=1}^{\infty} \cos \left( \frac{\theta}{2^k} \right) \]

from below
- $a_n \to$ same limit from above
- $\arctan(x) = \theta = \frac{\sin(\theta)}{\prod_{k=1}^{\infty} \cos \left( \frac{\theta}{2^k} \right)} = \frac{x}{B(a_0, g_0) \sqrt{1+x^2}}$
The Borchardt-Gauss-Carlson algorithm (BGC)

**BG computation of other inverse transcendentals is similar**
- same AGM recurrence as arctangent
- different initialization and postprocessing

**Convergence is linear or better**
- Original BG: contraction factor $\rightarrow \frac{1}{4}$ per iteration
- Carlson: Accelerate via Richardson extrapolation (BGC)
  - Contraction factor $\approx \frac{1}{1000}$ per iteration
  - Mildly superlinear: contraction factor improves per step
  - Some overhead, but no additional square roots
- Bottleneck is square root calculation
  - Use weak square root: compute $u = \sqrt{f}$ by solving $(v, u^2 - f) = 0$
Numerical results on the reals

Convergence of BGC arctangent and logarithm

- Very fast convergence over a large dynamic range
BGC has the features we’d like for PCE computation

<table>
<thead>
<tr>
<th>Good features</th>
</tr>
</thead>
<tbody>
<tr>
<td>■ Single algorithm applicable throughout domain of function</td>
</tr>
<tr>
<td>■ Needs only arithmetic plus square root</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Also</th>
</tr>
</thead>
<tbody>
<tr>
<td>■ Cost determined by square root cost</td>
</tr>
<tr>
<td>■ Needs $\sim 5$ square roots per computation</td>
</tr>
<tr>
<td>■ Scales well with dimension (equivalent to scaling of reciprocal)</td>
</tr>
</tbody>
</table>
All elementary inverse transcendentals can be expressed in terms of the Borchardt-Gauss mean.

The inverse transcendentals through the BG mean:

<table>
<thead>
<tr>
<th>Function</th>
<th>Relation to BG mean</th>
<th>Domain of applicability</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\tan^{-1}(x)$</td>
<td>$\frac{x}{B(1,\sqrt{1+x^2})}$</td>
<td>$-\infty &lt; x &lt; \infty$</td>
</tr>
<tr>
<td>$\sin^{-1}(x)$</td>
<td>$\frac{x}{B(\sqrt{1-x^2},1)}$</td>
<td>$-1 \leq x \leq 1$</td>
</tr>
<tr>
<td>$\cos^{-1}(x)$</td>
<td>$\frac{\sqrt{1-x^2}}{B(x,1)}$</td>
<td>$0 \leq x \leq 1$</td>
</tr>
<tr>
<td>$\tanh^{-1}(x)$</td>
<td>$\frac{x}{B(1,\sqrt{1-x^2})}$</td>
<td>$-1 &lt; x &lt; 1$</td>
</tr>
<tr>
<td>$\sinh^{-1}(x)$</td>
<td>$\frac{x}{B(\sqrt{1+x^2},1)}$</td>
<td>$-\infty &lt; x &lt; \infty$</td>
</tr>
<tr>
<td>$\cosh^{-1}(x)$</td>
<td>$\frac{\sqrt{x^2-1}}{B(x,1)}$</td>
<td>$x \geq 1$</td>
</tr>
<tr>
<td>$\log(x)$</td>
<td>$\frac{x-1}{B\left(\frac{x+1}{2},x\right)}$</td>
<td>$x &gt; 0$</td>
</tr>
</tbody>
</table>
Some theory

**Assuming no truncation of PCE**

- Neglecting truncation of PCE, convergence theory for BG is simple on any Banach space.
- On compact domain, $B(a_n, g_n)$ converges uniformly for any bounded $a_0, g_0$.
- As on $\mathbb{R}$, convergence is linear (factor 0.25, factor $\sim 10^{-3}$ with Carlson).

**Including truncation of PCE**

- Assuming truncation of PCE in square root, have proved weak convergence of $B(a_n, g_n)$.
Cost metrics for BGC iteration and line integration

Definitions

- $N_Q$ is number of quadrature points in line integration
- $N_{\text{Newt}}$ is number of Newton steps to compute square root
- $N_{BGC}$ is number of outer iterations in BGC method
- Cost of Newton step for a square root is 1 linear solve plus 1 symmetric matrix-matrix multiply ($\frac{5}{2}$ linear solve equivalents)

Cost required for each function (linear solve equivalents)

- **Arctangent**
  - Line integration: $N_Q$
  - BGC: $\frac{5}{2}(N_{BGC} + 1)N_{\text{Newt}} + 1$

- **Arcsine**
  - Line integration: $N_Q \left(\frac{5}{2}N_{\text{Newt}} + 1\right)$
  - BGC: $\frac{5}{2}(N_{BGC} + 1)N_{\text{Newt}} + 1$

- **Logarithm**
  - Line integration: $N_Q$
  - BGC: $\frac{5}{2}N_{BGC}N_{\text{Newt}} + 1$
Numerical results: $\log(x)$ on $[10^{-2}, 2]$

Error and cost for stochastic logarithm

- Errors shown for $p = 4, 12, 20$
- Line integration: number of linear solves grows with polynomial order
- BGC: number of linear solves independent of polynomial order
Numerical results: $\sin^{-1}(x)$ on $[0.1, 0.99]$

- Errors shown for $p = 4, 12, 20$
- Line integration and BGC have nearly identical accuracy
- BGC significantly less expensive than line integration
Numerical results: $\tan^{-1}(x)$ on $[-2, 9]$
Conclusions

The BGC algorithm is a promising approach to the fast and robust calculation of stochastic inverse transcendental functions:

- Rapid convergence
- Accuracy comparable to line integration
- Efficiency comparable to line integration for arctangent
- Efficiency superior to line integration for logarithm and arcsine

Implemented using Stokhos package of Trilinos library