# FAST COMPUTATION OF INVERSE TRANSCENDENTAL FUNCTIONS OF POLYNOMIAL CHAOS EXPANSIONS THROUGH ITERATED MEANS

KEVIN R. LONG, KALEB D. MCKALE, AND LOURDES JUAN

ABSTRACT. We present a new approach to the efficient computation of polynomial approximations to inverse transcendental functions of polynomials based on the modified arithmetic-geometric mean of Borchardt, Gauss, and Carlson. We carry out a systematic comparison with the line integration method of Debusschere *et al*. Our results indicate that this new method is essentially identical in accuracy to the line integration method. However, it is found to be significantly more efficient than line integration for computations of the arcsine, comparable in cost for the argtangent, while the relative efficiency of the new method and line integration on the logarithm is problem dependent.

## 1. INTRODUCTION

In the spectral approach to uncertainty quantification (*e.g.,*[7, 15]), the stochastic behavior of a system is approximated by a polynomial in the stochastic variables. Such a representation is often called a polynomial chaos expansion (PCE). In every intermediate stage of a calculation, the result is approximated by a PCE. The elementary operations of addition, subtraction, multiplication, and division involving truncated PCEs are easily carried out through closed-form calculations on the PCE coefficients; square roots (defined appropriately) may be computed very efficiently by Newton's method. However, computation of the elementary transcendental and inverse transcendental functions of PCEs is not such a simple matter. More precisely: it is the computation of PCE approximations to the elementary functions of PCEs that concerns us here. Hereafter, phrases such as "computing elementary functions of PCEs" should be understood as referring to computing *PCE approximations* to elementary functions of PCEs.

Debusschere *et al.* [5] have recently surveyed methods for computing elementary functions of PCEs. The usual methods of computing numerical values of elementary transcendental functions on $\mathbb{R}$ through piecewise polynomial approximations are usually not applicable for PCE computations, because such methods do not produce a single polynomial applicable over an interval (and thus usable in subsequent PCE calculations). Furthermore, the Taylor series for the inverse transcendentals have limited radius of convergence and therefore are not useful; a logarithm or arctangent, for instance, cannot be approximated by a Taylor polynomial in cases where the support of the stochastic variable's distribution extends beyond the radius of convergence. Computation by nonintrusive spectral projection (NISP) [6, 13] in more than a few stochastic dimensions requires the expensive approximation of high-dimensional integrals by cubature. Additionally, as we will

see below, NISP encounters even more critical difficulties when applied to certain of the inverse transcendentals with truncated arguments.

To sidestep these difficulties, [5] introduced a method based on integral and differential equation definitions of the elementary transcendentals. The forward transcendentals are defined by differential equations, the inverse transcendentals by integrals. For example, the arctangent of a PCE $u(\xi)$ is defined by the integral

$$\tan^{-1}(u(\xi)) = u(\xi)\int_0^1 \frac{dt}{1+t^2u(\xi)^2}. \tag{1.1}$$

This representation is exact; in practice, two approximations are made. First, the rational integrand is approximated by a PCE; second, the integration is done by quadrature with a PCE reciprocal done at each quadrature point. Though quadrature is still used, this integral is one-dimensional resulting in a considerable simplification over the multidimensional integrals encountered in NISP.

We introduce in this paper a quadrature-free approach to computation of inverse transcendentals of PCEs through iterated means. Though new to the world of PCE computation, the method of iterated means is a very old idea with important algorithms developed by Borchardt and Gauss in the 19th century and Archimedes in the 3rd century BCE[10]. The surprisingly rich history – dating back to Archimedes – of this and similar variants of iterated means is outlined in an expository article by Miel[10]. In the 1970s these algorithms were improved by Carlson[4] and Brent[3] for use in extended-precision calculation of inverse transcendentals on $\mathbb{R}$ and $\mathbb{C}$. At the heart of these methods lies the Borchardt-Gauss (BG) iteration

$$a_{n+1} = \frac{1}{2}(a_n + g_n) \tag{1.2a}$$

$$g_{n+1} = \sqrt{a_{n+1}g_n} \tag{1.2b}$$

with starting values $a_0 > 0$ and $g_0 > 0$. By appropriate choice of starting values and postprocessing steps, this basic BG iteration can be used to compute any of the elementary inverse transcendentals. This iteration is a variant of the better-known arithmetic-geometric mean (AGM) iteration used by Gauss for his development of the theory of elliptic integrals; the AGM has $\sqrt{a_n g_n}$ instead of $\sqrt{a_{n+1}g_n}$ in (1.2b). We will see that the BG iteration converges linearly with a contraction constant of $\frac{1}{4}$ and thus is too slow to be a practical method; however, Carlson showed that embedding the BG iteration in a simple Richardson extrapolation scheme acheives linear convergence with a contraction constant $\sim 10^{-3}$, sufficiently small to produce a practical method. We'll refer to the Borchardt-Gauss iteration as accelerated by Carlson's method as the BGC iteration.

Brent [3] has developed algorithms based on the AGM and asymptotic relations between the elliptic integrals and elementary functions that are quadratically convergent on the reals. These algorithms and their descendents are among the most efficient known for ultra high precision computations on the reals, and may be worth investigating for use with PCEs. In this paper we are concerned only with the BG and BGC algorithms.

Because the BG and BGC algorithms use only the arithmetic operations and the square root (all of which are efficiently computable on PCEs), and because (as will

be discussed below) the algorithm retains its accuracy over the entire real line, the potential for use in computation in a PCE setting is clear: one carries out the iterative procedure above not pointwise on a single real number, but on a PCE. However, simply porting an algorithm from $\mathbb{R}$ to a general Hilbert space does not necessarily work, so theoretical and practical issues need to be addressed. In particular, the usual convergence proofs for the BG iteration depends on properties specific to $\mathbb{R}$ or $\mathbb{C}$, so we will need to develop a convergence theory appropriate to PCEs in a Hilbert space. Furthermore, the square root operation must be handled carefully when working with truncated PCEs.

A preliminary investigation of this method was carried out by one of the authors (McKale) in an unpublished Master's thesis [9]. This paper extends that work with systematic numerical experiments and a first look at issues of convergence. We begin with a brief survey of the BG and BGC iterations on $\mathbb{R}$ and the connection to the inverse transcendentals. Next, we look at the question of an appropriate definition of the square root operation on PCEs, arguing that a weak definition of a square root should be used. We show that with this weak square root, the BG iteration applied to PCEs converges linearly in a weak sense, and furthermore, that Carlson's acceleration method is applicable to the PCE setting. Finally, we examine the asymptotic complexity of the BGC algorithm and the line integration method, and carry out numerical experiments to compare the efficiency and accuracy of these methods.

1.1. **Notation.** The formalism of polynomial chaos expansion is well described elsewhere[15, 7]. It is usually developed in the setting of probability, but for our purposes we can simply work in the setting of approximation of functions. Let $\Omega$ be a subset of $\mathbb{R}^D$. We use the symbol $\xi$ to represent coordinates on $\Omega$. We consider real-valued functions defined on $\Omega$, and introduce an inner product $(\cdot,\cdot)$ and $L^2$ norm $\|\cdot\|$. We then construct an orthogonal basis $\{\psi_i\}_{i=0}^{\infty}$ and introduce the triple product coefficients

$$c_{ijk} = \left(\psi_i, \psi_j\psi_k\right).$$

With the usual choice $\psi_0(\xi) = 1$ we have $c_{ij0} = 0$ when $i \neq j$. Furthermore, $c_{ijk}$ is invariant under permutation of indices.

We are interested in compositions $f \circ u$, where $f$ is an elementary inverse transcendental and $u(\xi)$ a polynomial. Though the methods of this paper are applicable to all seven inverse transcendentals (the inverse trigononometric and hyperbolic functions and the logarithm) we will concentrate primarily on three of the inverse transcendetals: the arctangent, the arcsine, and the logarithm. In the spectral approach to stochastic computation both the input to, and output of, a function $f$ are represented by infinite linear combinations of basis functions $\psi_i$,

$$u(\xi) = \sum_{i=0}^{\infty} u_i\psi_i(\xi) \tag{1.3}$$

$$f(u(\xi)) = \sum_{i=0}^{\infty} f_i\psi_i(\xi). \tag{1.4}$$

In practice, we work with sums that are truncated; for simplicity, we will assume throughout that the series for $u$ and $f$ are truncated at the same order $N$. We will refer to these series as polynomial chaos expansions.

1.2. **Current methods for inverse transcendentals.** We briefly review two current methods for computing inverse transcendentals of PCEs.

1.2.1. *Nonintrusive projection.* Perhaps the most straightforward approach to computing a spectral approximation to $f(u(\xi))$ is non-intrusive spectral projection (NISP). In this method, the generalized Fourier coefficients

$$f_i = c_{ii0}^{-1} \int_\Omega f(u(\xi)) \psi_i(\xi) \, d\mu \tag{1.5}$$

are computed by numerical integration; here $\mu$ represents a probability measure. Since the space $\Omega$ can have high dimension, the curse of dimensionality can make this a forbiddingly expensive calculation. However, a more fundamental concern for certain inverse transcendentals is the evaluation of the integrand itself. The difficulty is that the composition of $f$ with a *truncated* spectral representation of $u$ may not be computable throughout $\Omega$. If $f$ cannot be computed at all quadrature points, the NISP method fails.

Let $f : I_f \to \mathbb{R}$ be a real-valued function defined on an interval $I_f \subseteq \mathbb{R}$, and undefined outside that interval; the square root, logarithm, and arcsine are familiar examples (with domains of definition $[0, \infty)$, $(0, \infty)$, and $[-1, 1]$ respectively) that are relevant to the central topic of this paper. If we consider composition of $f$ with a function $u : \Omega \to I_f$, we encounter no difficulties as long as the method of computation of $u$ respects the restriction of its range to $I_f$. Unfortunately, spectral methods do not generally respect such restrictions.

As an illustrative example, consider the logarithm of the function

$$u(\xi) = \frac{1}{100} + \xi^8 \tag{1.6}$$

is clearly positive for all $\xi \in \mathbb{R}$; however, its best $L^2$ approximation (using the uniform inner product) on the quartics is

$$\hat{u}(\xi) = \frac{1}{42900} \left( 63000\xi^4 - 28000\xi^2 + 1929 \right) \tag{1.7}$$

which dips below zero on the interval $\approx [0.292, 0.599]$ and its reflection on the negative real line. Any NISP calculation having a quadrature point in that interval will fail.

Such examples are easily constructed, so this mode of failure is by no means restricted to isolated pathological cases. The NISP method cannot be considered a reliable method for functions with restricted domains such as the square root, logarithm, or arcsine.

| Function | Integral definition |
|---|---|
| $\tan^{-1}(u(\xi))$ | $u(\xi)\int_0^1 \frac{dt}{1+u(\xi)^2t^2}$ |
| $\sin^{-1}(u(\xi))$ | $u(\xi)\int_0^1 \frac{dt}{\sqrt{1-u(\xi)^2t^2}}$ |
| $\log(u(\xi))$ | $(u(\xi)-1)\int_0^1 \frac{dt}{1+(u(\xi)-1)t}$ |

TABLE 1. Summary of cost results for stochastic arctangent, arcsine, and logarithm computed by the line integration and BGC methods. The costs are reported in terms of the coefficient of $M^3$.

| Function | Relation to BG mean | Domain of applicability |
|---|---|---|
| $\arctan(x)$ | $\frac{x}{B\left(1,\sqrt{1+x^2}\right)}$ | $-\infty < x < \infty$ |
| $\arcsin(x)$ | $\frac{x}{B\left(\sqrt{1-x^2},1\right)}$ | $-1 \le x \le 1$ |
| $\arccos(x)$ | $\frac{\sqrt{1-x^2}}{B(x,1)}$ | $0 \le x \le 1$ |
| $\tanh^{-1}(x)$ | $\frac{x}{B\left(1,\sqrt{1-x^2}\right)}$ | $-1 < x < 1$ |
| $\sinh^{-1}(x)$ | $\frac{x}{B\left(\sqrt{1+x^2},1\right)}$ | $-\infty < x < \infty$ |
| $\cosh^{-1}(x)$ | $\frac{\sqrt{x^2-1}}{B(x,1)}$ | $x \ge 1$ |
| $\log(x)$ | $\frac{x-1}{B\left(\frac{x+1}{2},x\right)}$ | $x > 0$ |

TABLE 2. The inverse transcendental functions in terms of the Borchardt mean $B(\cdot,\cdot)$. Adapted from Miel [10].

1.2.2. *Line integration.* The line integral method of [5] computes inverse transcendentals in terms of their integral definitions, transformed to one dimensional integrals. The integrands are rational or algebraic functions of PCEs, which are approximated by truncated PCEs using algebraic operations described below. In practice the integrals are computed by quadrature, which means that those PCE operations must be performed at each quadrature point.

## 2. THE BORCHARDT-GAUSS ITERATION

The BG iteration for the inverse trigonometric functions can be understood in a simple way as recursive application of the half-angle identities, progressively reducing the argument so that a small-angle approximation becomes increasingly accurate. The reader is referred to Miel [10] or Acton [1] for an elementary derivation.

We establish some simple convergence results. The convergence theory for BG iteration on the reals is presented in Miel [10], and the most pertinent conclusions are summarized in theorem 1 below. However, we are interested in applying BG iteration to functions, and need to address the issues of convergence in norm and pointwise convergence.

**Theorem 1.** *We are given fixed real numbers $a_0 \geq 0$ and $g_0 > 0$. Then the iteration*

$$a_{n+1} = \frac{1}{2}(a_n + g_n) \tag{2.1}$$

$$g_{n+1} = \sqrt{g_n a_{n+1}} \tag{2.2}$$

*converges to the limit*

$$B(a_0, g_0) = \begin{cases} \frac{\sqrt{g_0^2 - a_0^2}}{\cos^{-1}(a_0/g_0)} & 0 \leq a_0 < g_0 \\ a_0 & a_0 = g_0 \\ \frac{\sqrt{a_0^2 - g_0^2}}{\cosh^{-1}(a_0/g_0)} & 0 < g_0 < a_0 \end{cases}. \tag{2.3}$$

*When $a_0 \neq g_0 \neq 0$, the sequence of iterates obeys one of the inequalities*

$$0 < g_n < g_{n+1} < a_{n+1} < a_n \quad (\textit{case } 0 < g_0 < a_0)$$
$$0 < a_n < a_{n+1} \leq g_{n+1} \leq g_n \quad (\textit{case } 0 < a_0 < g_0)$$

*Proof.* The verification of the inequalities follows easily from the iteration. For a proof of the limit, see [10]. □

Next, we look at convergence of BG iteration on functions. Our first result is that for suitable starting values defined on a compact domain, convergence is uniform.

**Theorem 2.** *Let $\Omega$ be a compact subset of $\mathbb{R}^D$, and suppose $\alpha_0 : \Omega \to \mathbb{R}$ and $\gamma_0 : \Omega \to \mathbb{R}$ are continuous functions that obey the inequalities $\alpha_0 \geq 0$, $\gamma_0 > 0$ throughout $\Omega$. Then the iteration*

$$\alpha_{n+1}(x) = \frac{1}{2}(\alpha_n(x) + \gamma_n(x)) \tag{2.4}$$

$$\gamma_{n+1}(x) = \sqrt{\gamma_n(x)\,\alpha_{n+1}(x)} \tag{2.5}$$

*converges uniformly to $B(\alpha_0(x), \gamma_0(x))$.*

*Proof.* From theorem 1 we have pointwise convergence of BG iteration to the limit $B(\alpha_0(x), \gamma_0(x))$ for every $x \in \Omega$. That limit is a continuous real-valued function throughout $\Omega$. Introduce the functions

$$L_n(x) = \min(\alpha_n(x), \gamma_n(x)) \tag{2.6}$$

$$U_n(x) = \max(\alpha_n(x), \gamma_n(x)) \tag{2.7}$$

and see from the inequalities in the conclusion of 1 that $L_n$ converges monotonically from below while $U_n$ converge monotonically from above. Uniform convergence follows from Dini's theorem [14]. □

Finally, the result on convergence rate easily extends from the reals to a Hilbert space.

**Theorem 3.** *Let $\Omega$ be a subset of $\mathbb{R}^D$, and suppose $\alpha_0 : \Omega \to \mathbb{R}$ and $\gamma_0 : \Omega \to \mathbb{R}$ are $L^2(\Omega)$ and obey the inequalities $\alpha_0 \geq 0$, $\gamma_0 > 0$ throughout $\Omega$. Let $\alpha_n$, $\gamma_n$ be iterates of 1.2a and 1.2b. Then the quantity $\left\|\alpha_n^2 - \gamma_n^2\right\|_{L^2} \to 0$ linearly with contraction constant $\frac{1}{4}$.*

*Proof.* Do an elementary calculation with the recurrence relations to find that

$$\alpha_{n+1}^2 - \gamma_{n+1}^2 = \frac{1}{4}\left(\alpha_n^2 - \gamma_n^2\right) \tag{2.8}$$

pointwise. □

While it is useful to have these theorems in hand, it's important to understand that they don't apply directly in practical PCE calculations. The theorems assume that the products and square roots are computed exactly, while in practical PCE computations they are always truncated. Indeed, with truncated PCE the conditions $\alpha_0 \geq 0$ and $\gamma_0 > 0$ may even be violated pointwise. A full study of convergence of truncated PCEs is beyond the scope of this exploratory paper. Finally, note that theorem 2 does not apply to PCEs set in non-compact domains such as the Hermite polynomial basis and Laguerre basis: we cannot expect uniform convergence of BG iteration in such cases (this question is distinct from the well-known issue of non-uniform convergence with polynomial degree).

In practice, we use not the BG iteration but the BGC iteration. As a Richardson extrapolation of the sequence of BG iterates, the limit is unchanged.

## 3. BORCHARDT-GAUSS ITERATIONS ON PCE

3.1. **Computation of the square root.** The square root is of fundamental importance in the BG and BGC algorithms. However, its computation on a space of PCEs is not entirely straightforward so the question of its robust and efficient calculation bears some discussion here. Consider $\sqrt{g}$ where $g : \Omega \to \mathbb{R}$ is nonnegative on $\Omega$, with PCE representation

$$g(\xi) = \sum_{i=0}^{\infty} g_i \psi_i(\xi). \tag{3.1}$$

The expansion coefficients $g_i$ are computed through orthogonal projection. In practice, however, the expansion is truncated at some maximum index $N$, and as seen above, this raises the possibility that the square root of the truncated polynomial will not exist pointwise. Therefore NISP is unsuitable for square root calculations.

To circumvent this difficulty we define the square root $\sqrt{\hat{g}}$ in a weak sense. Let $V^N$ be the span of the basis polynomials $\{\psi_i\}$. We define $\sqrt{\hat{g}}$ to be the real, nonnegative solution $r$ to

$$\left(v, r^2 - \hat{g}\right) = 0 \ \ \forall v \in V^N, \tag{3.2}$$

which need not require non-negativity of $\hat{g}$ at every point in $\Omega$. This is a system of nonlinear algebraic equations for the PCE coefficients $r_i$,

$$\sum_{j=0}^{N}\sum_{k=0}^{N} c_{ijk} r_j r_k - c_{ii} g_i = 0, \quad i = 0, 1, \cdots, N, \tag{3.3}$$

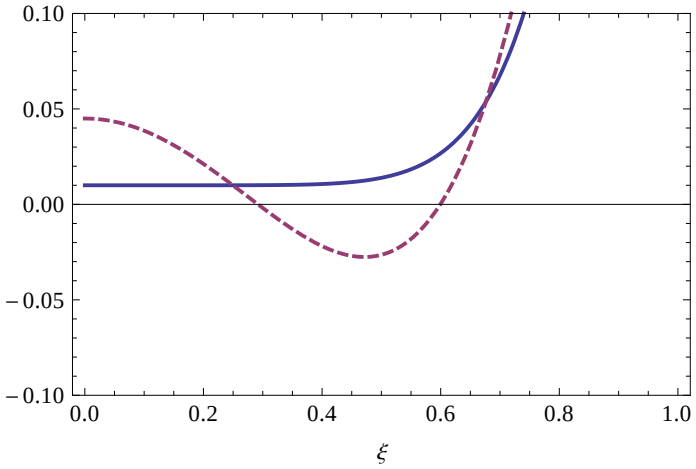


FIGURE 3.1. Comparison of exact $g(\xi)$ (solid line) and its quartic truncation $\hat{g}(\xi)$ (dashed line), zooming on a region near the horizontal axis to highlight the negative values resulting from truncation. The approximation was computed on the interval $[-1, 1]$, but only the positive half is shown here.

where $g_i$ are the PCE coefficients of $g$. This system can be solved approximately by Newton's method, with the step $\delta r_j$ given by the linear equation

$$\sum_{j=0}^{N}\sum_{k=0}^{N} c_{ijk} r_k^{(n)} \left( r_j^{(n)} + 2\delta r_j \right) + c_{ii0} g_i = 0. \tag{3.4}$$

Discussion of existence of weak square roots real solutions to 3.3 is in order. Ideally, we would like a theorem ensuring that a best $L^2$ polynomial approximation to any nonnegative function will have a weak square root that is real and nonnegative; however, we know of know such theorem. A theorem by Barvinok [2] shows that there is an algorithm, polynomial time in $N$, for determining whether systems of quadratic equations such as3.3 have real solutions. An open question in real a;gebraic geometry is whether the highly structured form of 3.3, in which the coefficients $c_{ijk}$ are known to be triple products of orthogonal polynomials, can be exploited to provide an existence theorem for weak square roots.

Though we have to date been unable to prove existence of real, nonnegative weak square roots, in none of our experiments were failures to solve 3.3 found.

3.2. **Cost estimates.** The cost of a BGC iteration is determined by the cost of the three operations involved: addition, multiplication, and the square root. The line integration method uses division in addition to these, so looking ahead to comparison between BGC and line integration we estimate the cost of division as well.

We'll assume dense linear algebra throughout, and disregard all but the leading order terms. Let $M = N + 1$ b the number of PCE coefficients in the expansion of

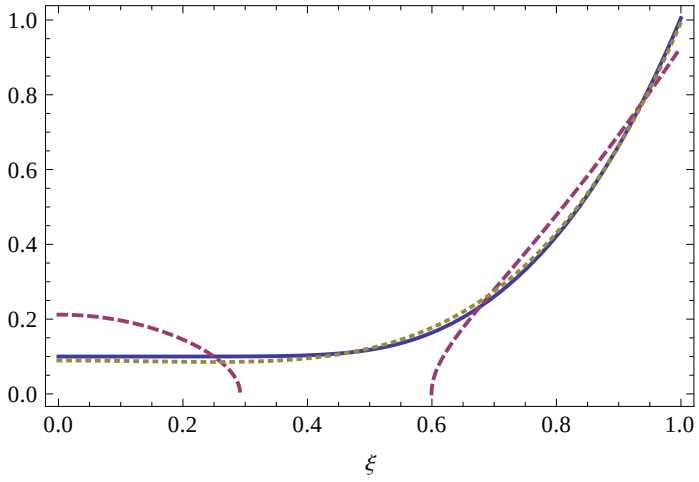


FIGURE 3.2. Comparison of the exact square root of $g(\xi)$ (defined in equation 1.6)and two methods of approximation. The exact square root of the exact $g$ is shown with the solid line. The pointwise square root $\sqrt{\hat{g}}$ is shown with the dashed line; it fails to exist in the interval where the truncated approximation $\hat{g}$ is negative. The weak square root is shown with the dotted line.

$\hat{g}$ and its weak square root $\hat{r}$. At the $n$-th Newton step, the residual vector

$$c_{ii0}g_i + \sum_{j=0}^{N}\sum_{k=0}^{N} c_{ijk}r_j^{(n)}r_k^{(n)} \tag{3.5}$$

must be formed, at a cost $\frac{3}{2}M^3$ flops. The Jacobian

$$2\sum_{k=0}^{N} c_{ijk}r_k^{(n)} \tag{3.6}$$

is formed at each Newton step at a cost $2M^3$ flops. The cost of a solve is dominated by factorization at $\frac{2}{3}M^3$ flops, so each Newton step costs $\frac{25}{6}M^3 \approx 4M^3$ flops.

The costs of the operations of addition, subtraction, multiplication, and division of PCEs should also be computed. Addition and subtraction are both $O(M)$ and can be considered free compared to the other operations. Multiplication of two PCEs is done as

$$(uv)_i = \sum_{j=0}^{N}\sum_{k=0}^{N} c_{ijk}u_j v_k \tag{3.7}$$

and requires $3M^3$ flops (or half that for the important case $u = v$ where symmetry can be exploited). Division of two PCEs $q = u/v$ is done implicitly by forming and solving the system

$$\sum_{j=0}^{N}\sum_{k=0}^{N} c_{ijk}q_j v_k = c_{ii0}u_i \tag{3.8}$$

| Operation | Cost $(M^3)$ |
|---|---|
| Addition/subtraction | 0 |
| Multiplication | 3 ($\frac{3}{2}$ for a square) |
| Division | $\frac{8}{3}$ |
| Square root | $\frac{25}{6}N_{Newt}$ |

TABLE 3. Costs of the four arithmetic operations and the square root in units of $M^3$, where $M$ is the number of PCE coefficients and $N_{Newt}$ is the number of Newton iterations needed in a square root calculation. In the square root cost estimate, the exact coefficient $\frac{25}{6}$ is approximated by 4.

| Function | Integral definition | LI cost / $M^3$ | BGC cost / $M^3$ |
|---|---|---|---|
| $\tan^{-1}(u(\xi))$ | $u(\xi)\int_0^1 \frac{dt}{1+u(\xi)^2t^2}$ | $3+\frac{25}{6}N_Q$ | $\frac{5}{2}(N_{BGC}+1)N_{Newt}+1$ |
| $\sin^{-1}(u(\xi))$ | $u(\xi)\int_0^1 \frac{dt}{\sqrt{1-u(\xi)^2t^2}}$ | $3+\frac{25}{6}(1+N_{Newt})N_Q$ | $\frac{5}{2}(N_{BGC}+1)N_{Newt}+1$ |
| $\log(u(\xi))$ | $(u(\xi)-1)\int_0^1 \frac{dt}{1+(u(\xi)-1)t}$ | $3+\frac{8}{3}N_Q$ | $\frac{5}{2}N_{BGC}N_{Newt}+1$ |

TABLE 4. Summary of cost results for stochastic arctangent, arcsine, and logarithm computed by the line integration and BGC methods. The costs are reported in terms of the coefficient of $M^3$. $N_Q$ is the number of quadrature points, and $N_{\text{Newt}}$ is the number of Newton iterations needed in the computation of the weak square root.

for $q_j$. Forming the system requires $2M^3$ flops, and solving it takes $\frac{2}{3}M^3$ flops for a total cost of $\frac{8}{3}M^3$.

3.2.1. *Cost of line integration.* For comparison to BGC, we also provide cost estimates for the line integration method. The cost will depend on the number of quadrature points required. The arctangent requires a square ($\frac{3}{2}M^3$) and a reciprocal ($\frac{8}{3}M^3$) at each quadrature point, plus one multiply $(3M^3)$ after summation. The arcsine requires a square $(\frac{3}{2}M^3)$, a reciprocal $(\frac{8}{3}M^3)$, and a square root $(\frac{25}{6}N_{Newt}M^3)$ at each quadrature point, plus one multiply $(3M^3)$ after summation. The logarithm requires a reciprocal $(\frac{8}{3}M^3)$ at each quadrature point, plus one multiply $(3M^3)$ after summation.

The costs of BCG iteration and line integration (LI) are summarized in table 4. The actual costs must be measured experimentally, because we cannot predict in advanced the number of quadrature points for line integration, number of Newton iterations for the square root, and number of BGC iterations needed for convergence.

## 4. NUMERICAL EXPERIMENTS

The BGC and line integration algorithms were implemented in C++ using the Stokhos [11] package of the Trilinos [8] suite. The Stokhos square root function

is implemented using NISP, so we wrote a replacement square root the computes the weak square root with Newton's method. Our software implementation was instrumented to accumulate the cost of the elementary operations on PCEs based on the asymptotic costs estimates for each elementary operation.

A series of experiments was designed to compare the accuracy and efficiency of the line integration and BGC methods as applied to the logarithm, arctangent, and arcsine. In all experiments, the only distribution considered is the uniform distribution on $[-1, 1]$. The basis functions are the Legendre polynomials.

In each experiment, the BGC iteration is compared to two implementations of the line integration method, differing by virtue of different quadrature methods. The first line integration method, which we'll call "adaptive," uses an adaptive $7/15$ point Gauss-Kronrod method [12] with recursive local subsectioning. This is representative of many simple accuracy-controlled black box methods. The second line integration method, which we'll call "prescient," tries a sequence of Gauss-Legendre rules with increasing order until a desired tolerance is reached, and then records only the cost of the final rule. By ignoring the cost of the trial integrations, the cost of our "prescient" method represents the cost an integration procedure in which a user has sufficient prescience to choose in advance, for each and every problem, exactly the right Gauss-Legendre rule to reach a desired accuracy with minimum work. The logic behind our choice of these two methods is that an adaptive scheme tailored to these problems can no doubt be more efficient than our simple adaptive method but is unlikely to do quite as well as the "prescient" method. Therefore, we consider that a good implementation of the line integration method will lie somewhere between these two methods in efficiency.

In the first set of experiments, outlined in table 5, functions are evaluated on four intervals whose endpoints are listed in the table. Experiments $A_n$ involve the logarithm, $B_n$ the arctangent, and $C_n$ the arcsine. For each function, the intervals are chosen to range from unchallenging (interval size small compared to the curvature of the function) to challenging (large interval size in the case of the arctangent, or approaching a domain endpoint in the case of the logarithm or arcsine). Mapped back to the standard interval $[-1, 1]$ for the uniform distribution, we can consider each of these experiments to apply to a composition of an inverse transcendental with a first degree polynomial.

In the second set of experiments, compositions with nonlinear functions are considered. For each of the three inverse transcendentals under study, a simple family of nonlinear functions is listed in table 6. The functions are chosen to produce curves that are difficult to fit with low-degree polynomials, and in the case of the logarithm and arcsine, to push close to a domain boundary.

In all experiments, the three algorithms (BGC, LIM/adaptive, and LIM/prescient) were run with two convergence tolerances: $\tau = 10^{-5}$ and $\tau = 10^{-10}$. For each case, the polynomial degree was varied from 4 to 32 in steps of 2. For each case, we plot the following: (a) $L^2$ norm of the error versus PCE degree, (b) computational cost versus PCE degree.

4.1. **Accuracy and convergence.** The $L^2$ error is plotted against PCE degree for experiments $A$, $B$, and $C$ in figures 4.1 and 4.2 (convergence tolerance $10^{-5}$) and in

| Experiment ID / $n$ | $A_n$ | $B_n$ | $C_n$ |
|---|---|---|---|
| 1 | $[0.9, 1]$ | $[0.9, 1.1]$ | $[0.4, 0.6]$ |
| 2 | $[0.5, 1]$ | $[0.5, 1.5]$ | $[-0.5, 0.5]$ |
| 3 | $[0.1, 1]$ | $[-1, 2]$ | $[-0.5, 0.9]$ |
| 4 | $[0.01, 1]$ | $[-5, 10]$ | $[-0.5, 0.99]$ |
| Function | $\log$ | $\tan^{-1}$ | $\sin^{-1}$ |

TABLE 5. Experiments with linear arguments. In this set, cases labeled with higher indices have intervals that are more computationally challenging.

| Experiment ID | Function |
|---|---|
| $D_p$ | $\log\left(\frac{1}{100} + x^p\right)$ |
| $E_p$ | $\tan^{-1}\left(4 - 8x^p\right)$ |
| $F_p$ | $\sin^{-1}\left(\frac{99}{100} - x^p\right)$ |

TABLE 6. Experiments with nonlinear arguments. Cases with $p = 2$ and $p = 8$ were run.

figures 4.3 and 4.4 (convergence tolerance $10^{-10}$). The results indicate that the BGC and line integration methods are comparable in accuracy; the error curves differ significantly only after the desired tolerance has been exceeded. The adaptive quadrature method used seems to overconverge, systematically producing higher accuracy than requested. All three methods converge superlinearly with degree, until desired tolerance is reached.

Similar results are found for experiments $D$,$E$,$F$, as shown in figures 4.5 (tolerance $10^{-5}$) and 4.6 (tolerance $10^{-10}$). Again, convergence is superlinear until the desired tolerance is reached.

Examples of BGC approximations (tolerance $10^{-5}$) in experiments $D$,$E$,$F$ are shown in figure 4.7. As can be seen, the difficult feaures such as cusps and plateaus are fit well.

4.2. **Cost measurements.** Our software implementation was instrumented to accumulate the cost of the elementary operations on PCE based on the asymptotic costs estimates for each elementary operation. We chose this approach over timings in order to compare efficiency of algorithms rather than efficiency of particular software implementations.

While the BGC and line integration methods are comparable in accuracy, they are quite different in efficiency. Which method is more efficient depends on the function considered. Our cost estimates suggest a clear efficiency advantage for the BGC in the computation of arcsines, but no clear difference for the arctangent and logarithm. For accurate cost computations, the actual number of Newton steps and quadrature points required were measured in our experiments. As expected, in all arcsine experiments it was found that BGC iteration was more efficient than

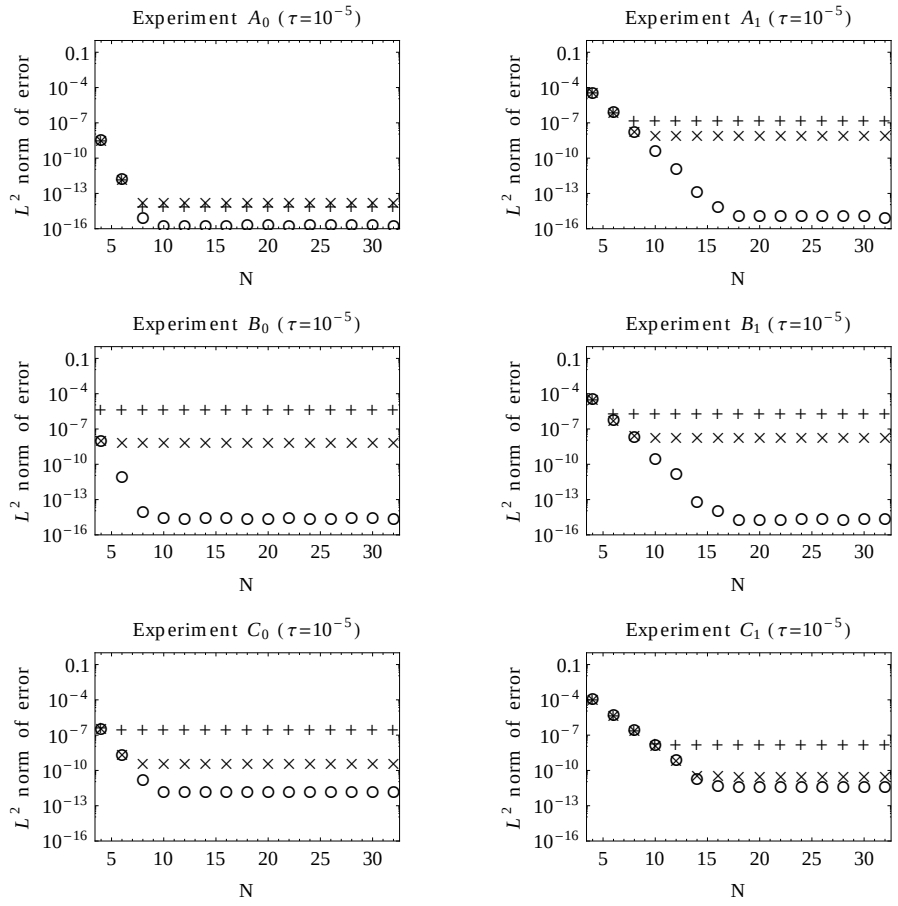


FIGURE 4.1. Convergence results for experiment set ABC, intervals 0 and 1, convergence tolerance $\tau = 10^{-5}$. Refer to table 5 for definitions of the functions used in these experiments. Results from BGC iteration, adaptive quadrature, and prescient quadrature are shown with crosses ($\times$), circles ($\circ$), and plus signs (+) respectively.

line integration. Results for the arctangent generally favor line integration. In no case was BGC computation of the arctangent more efficient than line integration with "prescient" quadrature, though with only a few exceptions (on very simple problems) it was more efficient than line integration with black-box adaptive quadrature. Results for the logarithm were mixed, with prescient line integration more efficient than BGC on the easier (more nearly linear) problems, and BGC more efficient on the more nonlinear problems.

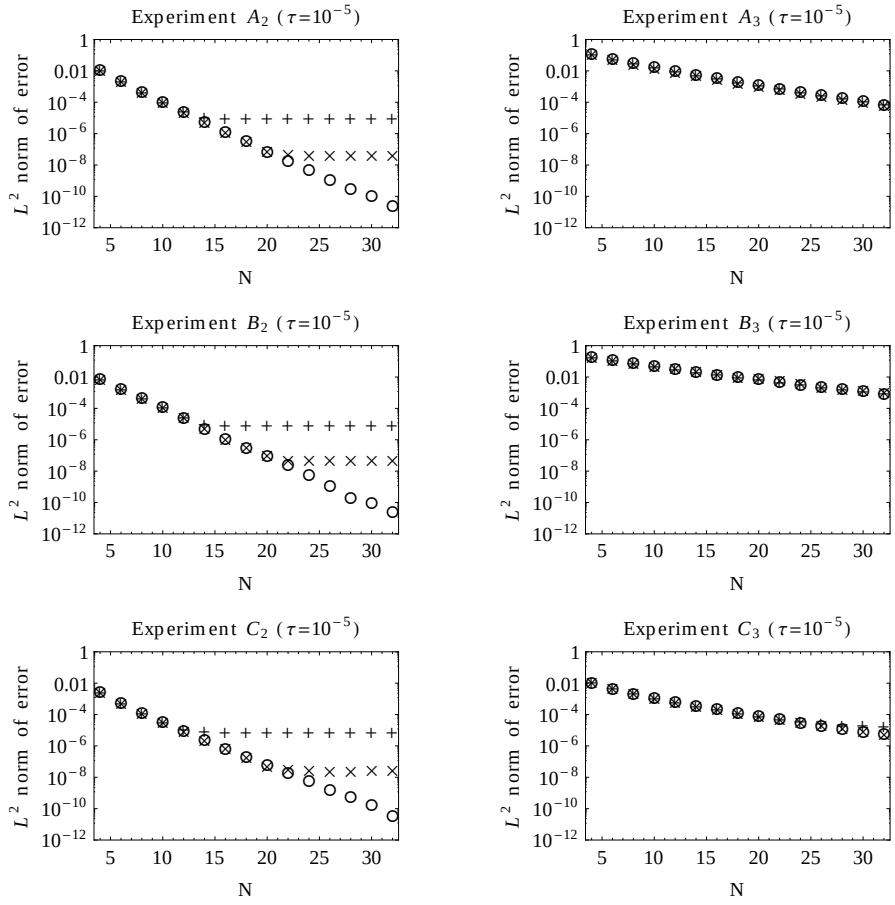


FIGURE 4.2. Convergence results for experiment set ABC, intervals 2 and 3, convergence tolerance $\tau = 10^{-5}$. Refer to table 5 for definitions of the functions used in these experiments. Results from BGC iteration, adaptive quadrature, and prescient quadrature are shown with crosses ($\times$), circles ($\circ$), and plus signs (+) respectively.

## 5. CONCLUSIONS

We have investigated the application of the Borchardt-Gauss-Carlson method of iterated means to the computation of inverse transcendental functions of polynomials. This method provides an approximation that is accurate over a wide range of argument, and furthermore bypasses difficulties with restricted domain that appear when applying non-intrusive projection to compositions of some inverse transcendentals with polynomial approximations. We have analyzed the convergence of the method only in the simplified case in which truncation of the polynomial approximations is ignored.

The cost of the method is dominated by the need to compute a square root of a PCE at each iteration. With Carlson's acceleration, we find experimentally that

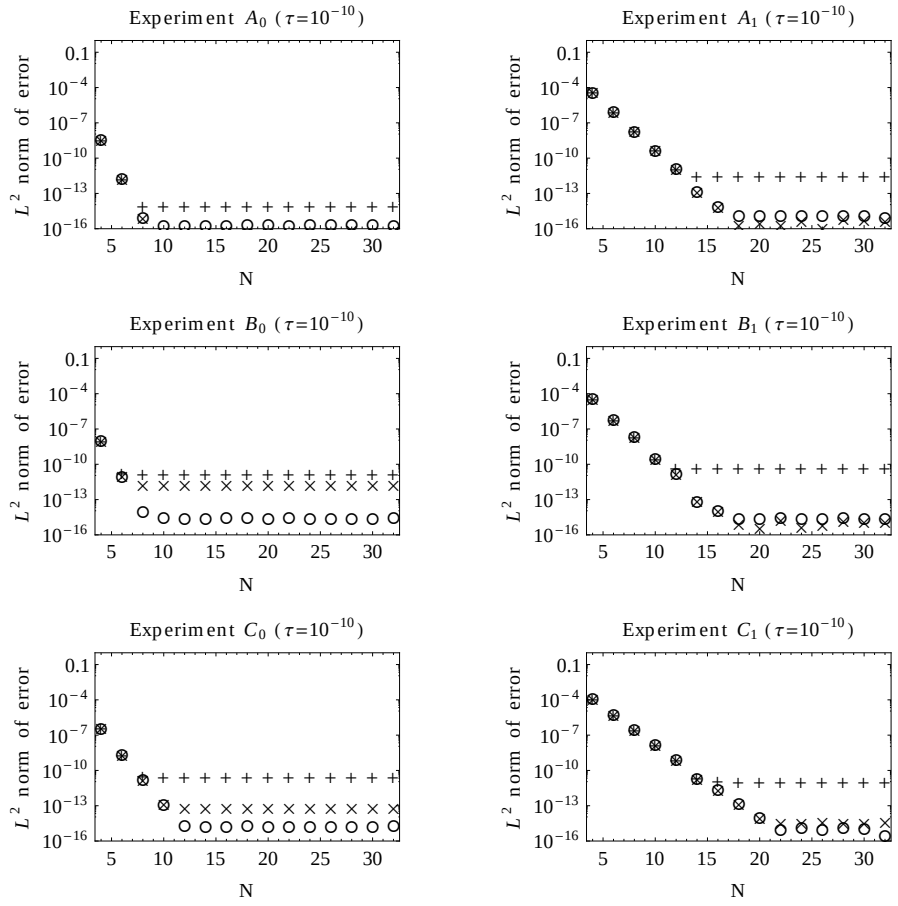


FIGURE 4.3. Convergence results for experiment set ABC, intervals 0 and 1, convergence tolerance $\tau = 10^{-10}$. Refer to table 5 for definitions of the functions used in these experiments. Results from BGC iteration, adaptive quadrature, and prescient quadrature are shown with crosses ($\times$), circles ($\circ$), and plus signs (+) respectively.

each iteration reduces the error by about $10^{-3}$, so only a few iterations are needed for acceptable accuracy. We find that for a specified convergence tolerance in computation of the coefficients, the accuracy of the PCE coefficients computed by BGC iteration and by line integration are nearly identical, with no systematic favoring of either method.

Based on our timing results, we recommend that BGC iteration be considered the method of choice for the arcsine. The reason is the square root in the integrand, making each quadrature point roughly as expensive as a single BGC step. We anticipate that this conclusion will pertain to all four of the elementary transcendentals that involve square roots in their integral definitions: arcsine, arccosine, arcsinh, and arccosh. We note also that improvement in linear and nonlinear

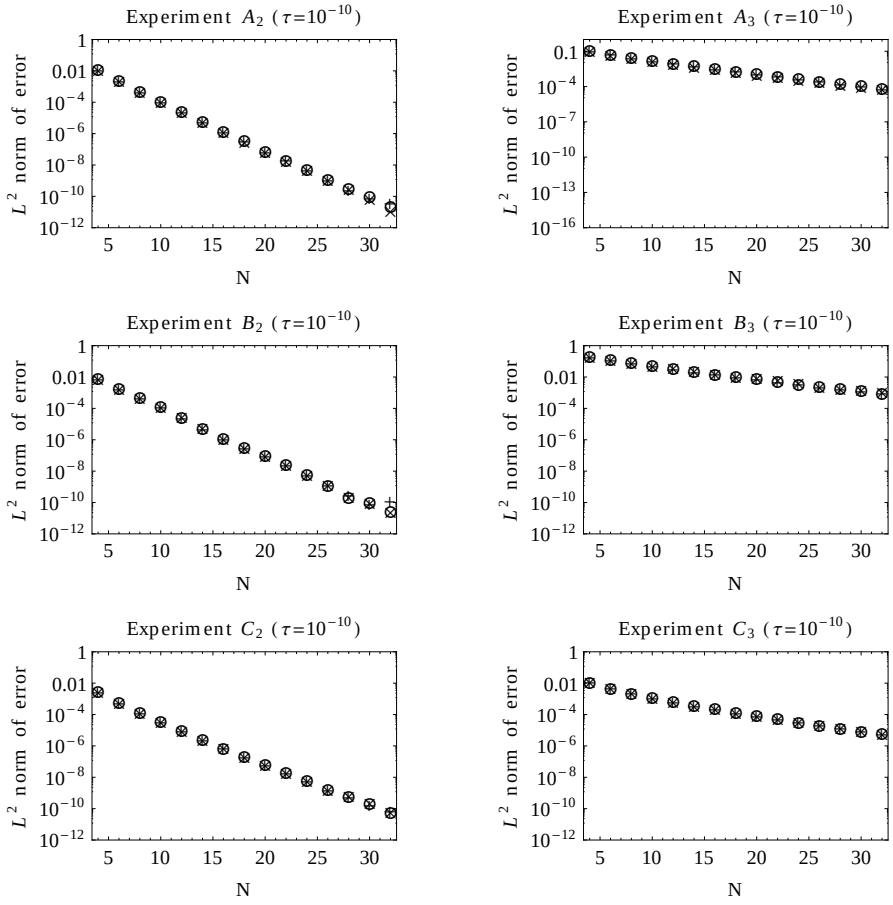


FIGURE 4.4. Convergence results for experiment set ABC, intervals 2 and 3, convergence tolerance $\tau = 10^{-10}$. Refer to table 5 for definitions of the functions used in these experiments. Results from BGC iteration, adaptive quadrature, and prescient quadrature are shown with crosses ($\times$), circles ($\circ$), and plus signs (+) respectively.

solver algorithms for square root computation will impact both classes of methods equally.

For the arctangent, BGC iteration is systematically less efficient than line integration with our artificially prescient quadrature method, though it usually outperforms line integration with a black-box adaptive algorithm. Whether BGC or line integration should be used will apparently depend on whether suitable quadrature routines can be found.

For the logarithm, neither BGC iteration or line integration is clearly superior in all circumstances. Further investigation is needed to devise an adaptive algorithm that will choose a method based on properties of the argument.

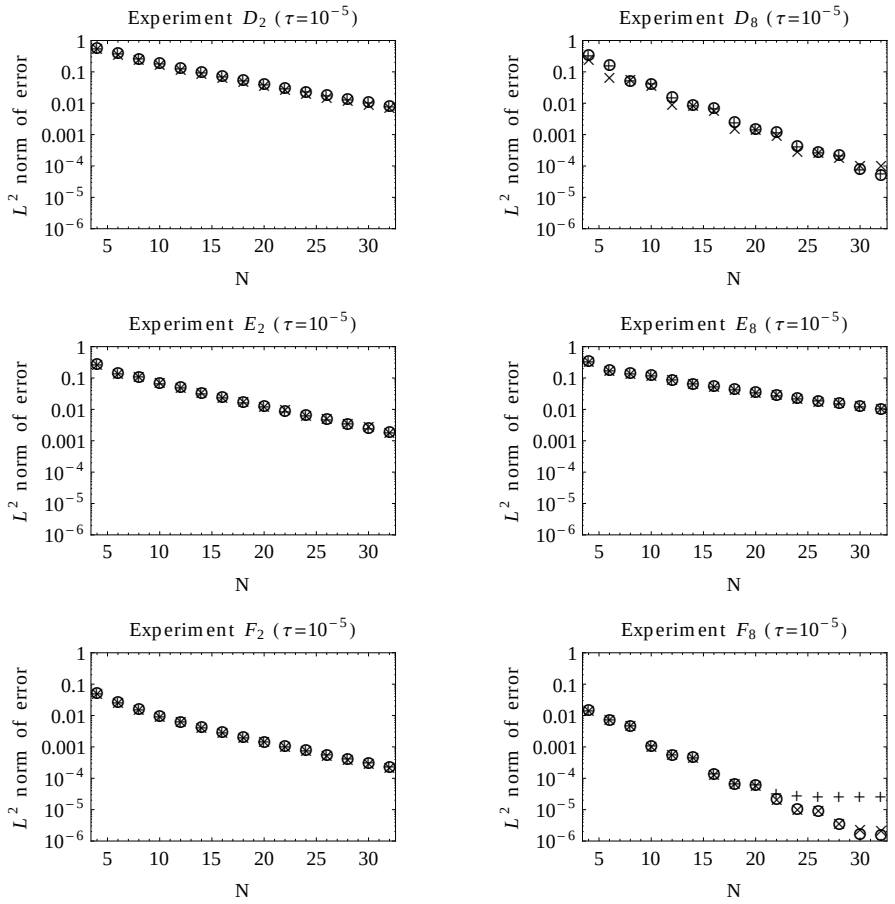


FIGURE 4.5. Convergence of $L^2$ norm of error with polynomial order $N$ in experiments D,E,F with $p = 2$ and 8 and tolerance $\tau = 10^{-5}$. Refer to table 6 for definitions of the functions used in these experiments. Results from BGC iteration, adaptive quadrature, and prescient quadrature are shown with crosses ($\times$), circles ($\circ$), and plus signs (+) respectively.

Finally, we note that iterated means are also applicable to certain special functions such as elliptic integrals and certain hypergeometric functions. This class of methods may prove useful for PCE approximations to those functions as well as the elementary inverse transcendentals.

5.1. **Acknowledgements.** The authors would like to thank Dr. Eric Phipps of Sandia National Laboratories for his assistance with the Stokhos library. This work was supported by US Department of Energy Advanced Scientific Computing program, award DE SC001301, and the US National Science Foundation, award 0904834.

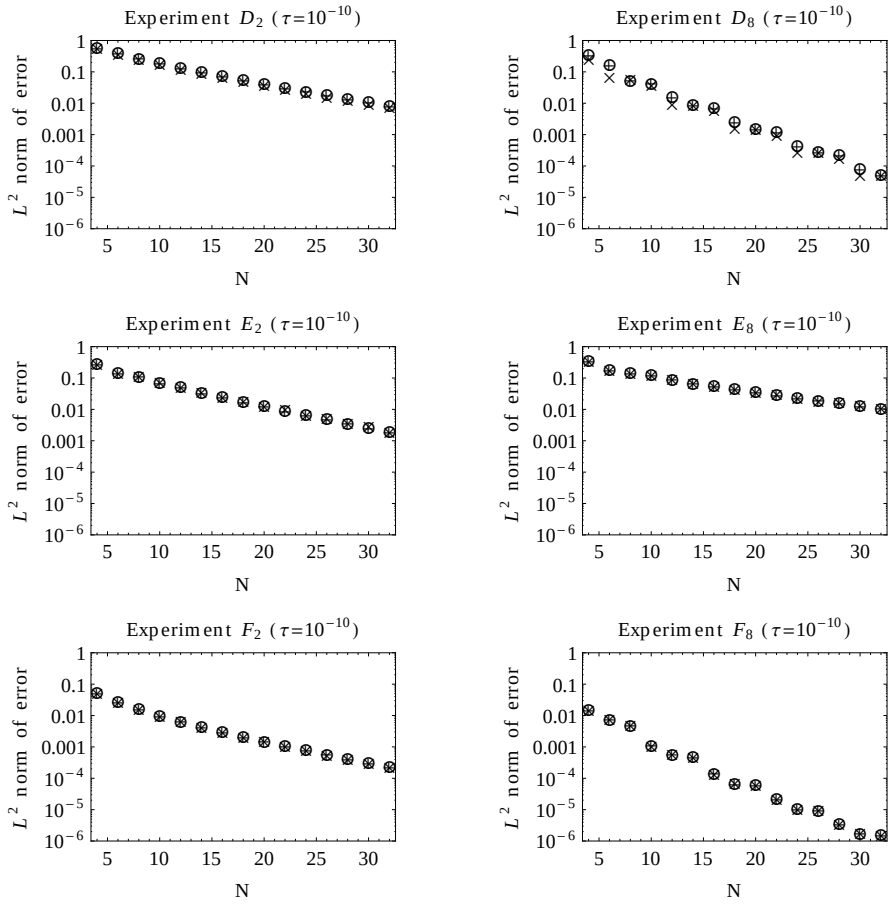


FIGURE 4.6. Convergence of $L^2$ norm of error with polynomial order $N$ in experiments D,E,F with $p = 2$ and 8 and tolerance $\tau = 10^{-10}$. Refer to table 6 for definitions of the functions used in these experiments. Results from BGC iteration, adaptive quadrature, and prescient quadrature are shown with crosses ($\times$), circles ($\circ$), and plus signs (+) respectively.

## REFERENCES

1. Forman S. Acton, *Numerical Methods That Work*, Mathematical Association of America, Washington D.C., 1990.
2. A. I. Barvinok, *Feasibility testing for systems of real quadratic equations*, Discrete and Computational Geometry **10** (1993), no. 1, 1–13.
3. Richard P. Brent, *Fast Multiple-Precision Evaluation of Elementary Functions*, Journal of the ACM **23** (1976), no. 2, 242–251.
4. B. C. Carlson, *An algorithm for computing logarithms and arctangents*, Mathematics of Computation **26** (1972), no. 118, 543–543.
5. Bert J. Debusschere, Habib N. Najm, Philippe P. Pébay, Omar M. Knio, Roger G. Ghanem, and Olivier P. Le Maître, *Numerical challenges in the use of polynomial chaos representations for stochastic processes*, SIAM J. Sci. Comput. **26** (2005), no. 2, 698–719.

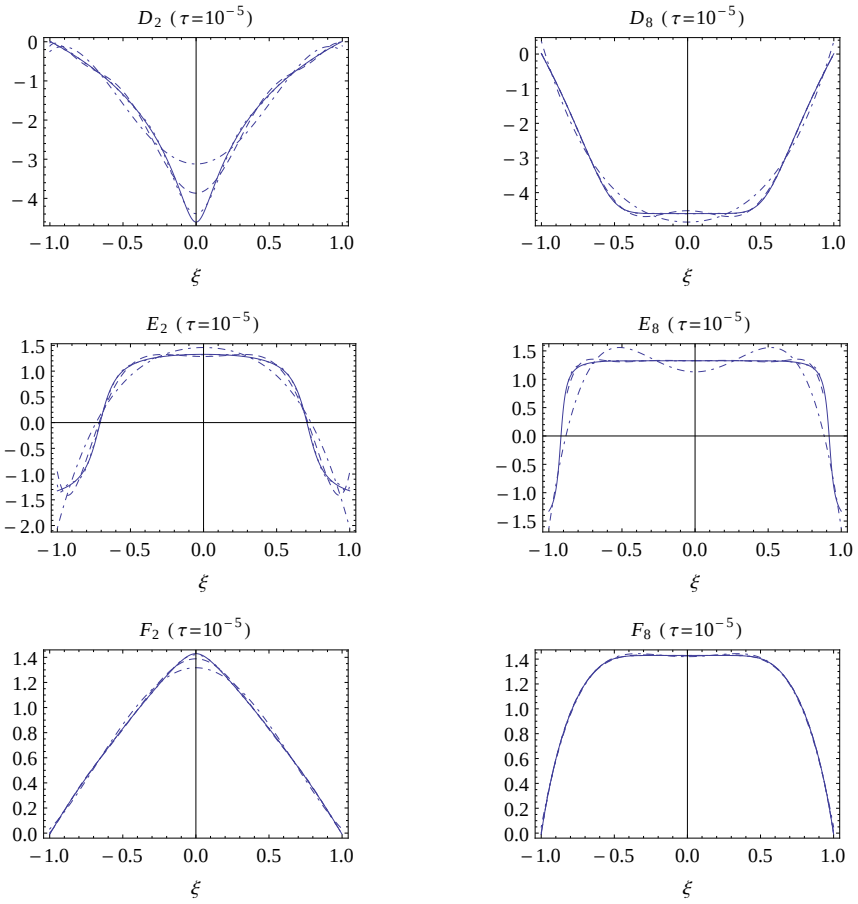


FIGURE 4.7. BGC approximations of degree $N = 4, 8, 16$ to the functions of experiments $D_p$, $E_p$, and $F_p$ with $p = 2$ and 8 and tolerance $\tau = 10^{-5}$. Exact functions are shown in solid lines. Approximations with $N = 4, 8, 16$ are shown with dot-dashed, dashed, and dotted lines.


6. R Ghanem, J Red-Horse, and A Sarkar, *8th asce specialty conference of probabilistic mechanics and structural reliability*, ASCE, 2000.
7. R. G. Ghanem and P. D. Spanos, *Stochastic finite elements: A spectral approach*, Dover, 2012.
8. Michael A. Heroux, Roscoe A. Bartlett, Vicki E. Howle, Robert J. Hoekstra, Jonathan J. Hu, Tamara G. Kolda, Richard B. Lehoucq, Kevin R. Long, Roger P. Pawlowski, Eric T. Phipps, Andrew G. Salinger, Heidi K. Thornquist, Ray S. Tuminaro, James M. Willenbring, Alan Williams, and Kendall S. Stanley, *An overview of the trilinos project*, ACM Trans. Math. Softw. **31** (2005), no. 3, 397–423.
9. K.D. McKale, *Archimedes, Gauss and stochastic computation: A new (old) approach to fast algorithms for the evaluation of transcendental functions of generalized polynomial chaos expansions*, Master's thesis, Texas Tech University, 2011.
10. G. Miel, *Of calculations past and present: The Archimedean algorithm*, The American Mathematical Monthly **90** (1983), 17–35.
11. R P Pawlowski, E.T. Phipps, and A G Salinger, *Automating embedded analysis capabilities and managing software complexity in multiphysics simulation, Part I: Template-based generic programming*, Scientific Programming **20** (2012), 197–219.

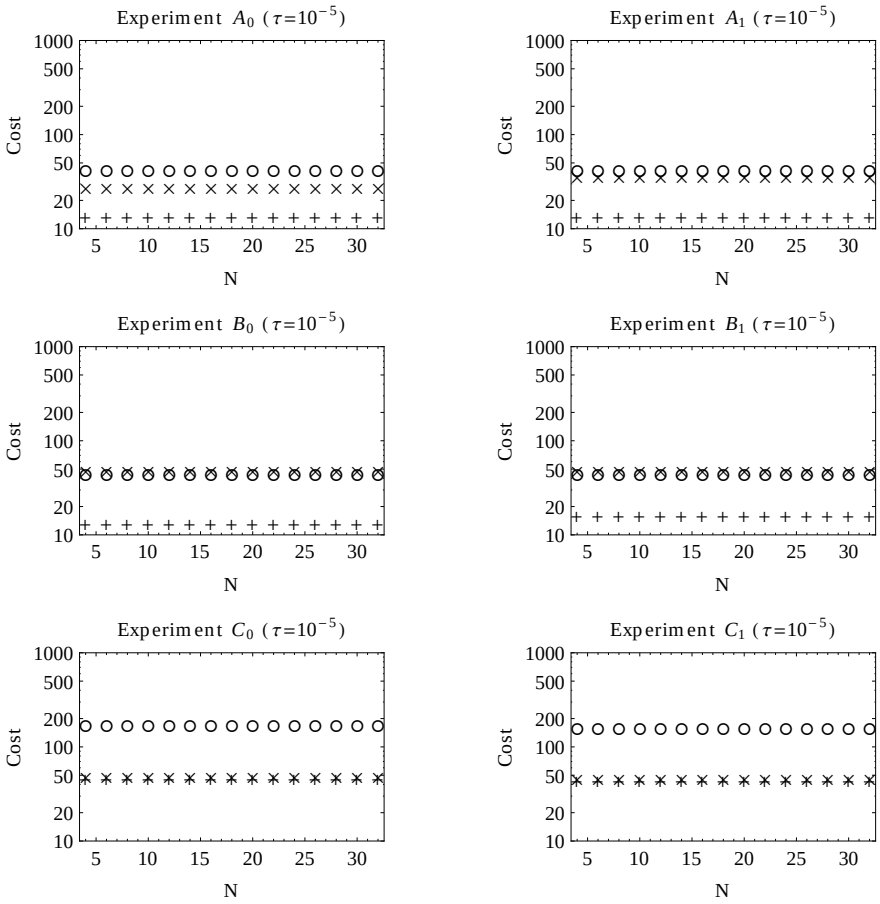


FIGURE 4.8. Cost results from experiment set ABC, intervals 0 and 1, convergence tolerance $\tau = 10^{-5}$. Results from BGC iteration, adaptive quadrature, and prescient quadrature are shown with crosses ($\times$), circles ($\circ$), and plus signs (+) respectively.

12. R. Piessens, E. de Doncker-Kapenga, C. W. Uberhuber, and D. K. Kahaner, *QUADPACK, a subroutine package for automatic integration*, Springer-Verlag, 1983.
13. Matthew T Reagan, Habib N Najm, Roger G Ghanem, and Omar M Knio, *Uncertainty quantification in reacting-flow simulations through non-intrusive spectral projection*, Combustion and Flame **132** (2003), no. 3, 545–555.
14. H.L. Royden and P.M. Fitzpatrick, *Real Analysis*, 4th ed., Prentice Hall, Boston, 2010.
15. D. Xiu, *Numerical methods for stochastic computations: A spectral method approach*, Princeton, 2010.

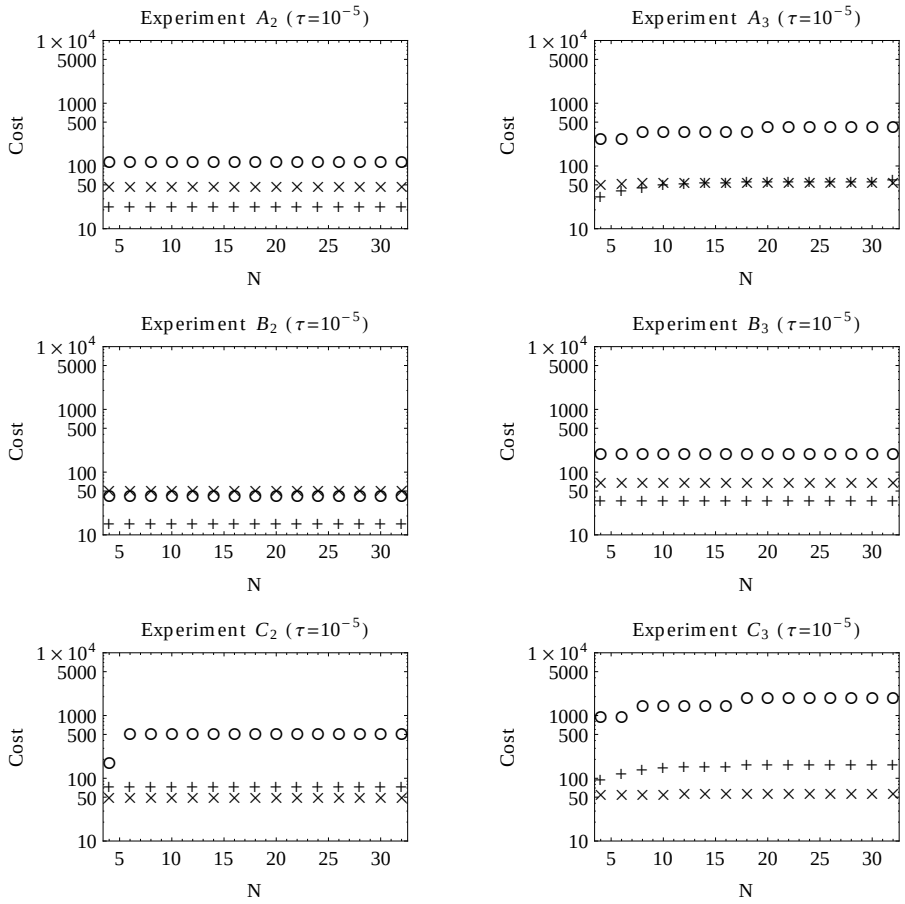


FIGURE 4.9. Cost results from experiment set ABC, intervals 2 and 3, convergence tolerance $\tau = 10^{-5}$. Results from BGC iteration, adaptive quadrature, and prescient quadrature are shown with crosses ($\times$), circles ($\circ$), and plus signs (+) respectively.

DEPARTMENT OF MATHEMATICS AND STATISTICS, TEXAS TECH UNIVERSITY

*E-mail address*: kevin.long@ttu.edu

DEPARTMENT OF MATHEMATICS AND STATISTICS, TEXAS TECH UNIVERSITY

*E-mail address*: kaleb.mckale@ttu.edu

DEPARTMENT OF MATHEMATICS AND STATISTICS, TEXAS TECH UNIVERSITY

*E-mail address*: lourdes.juan@ttu.edu

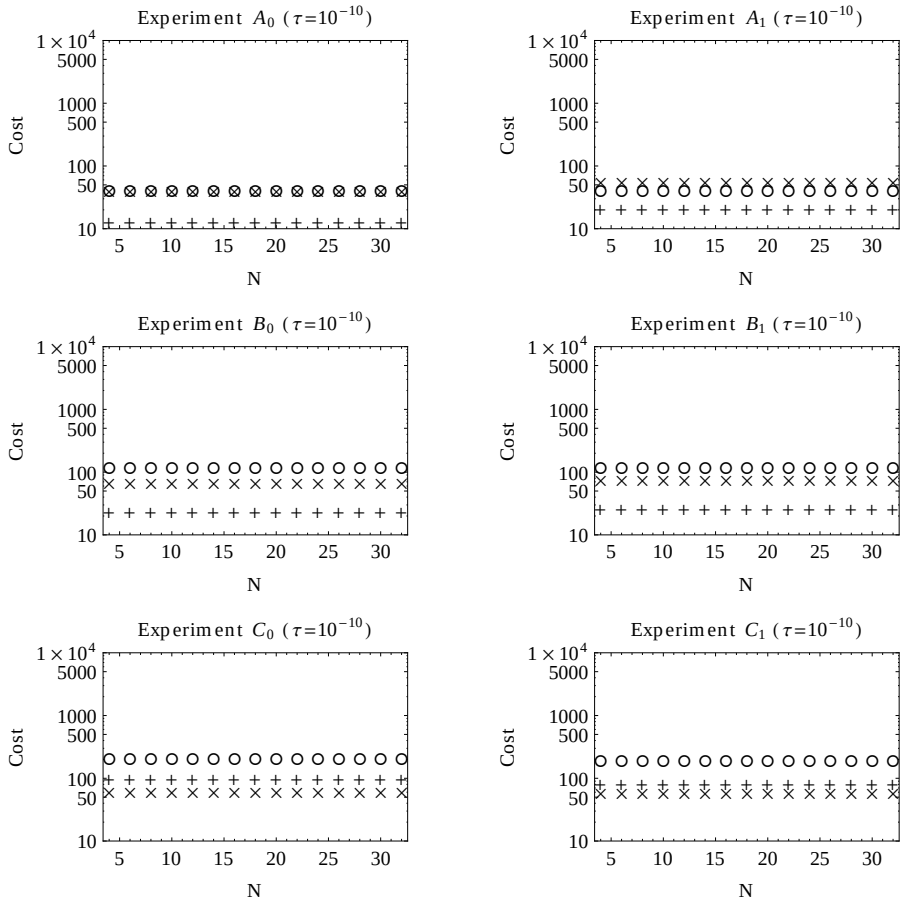


FIGURE 4.10. Cost results from experiment set ABC, intervals 0 and 1, convergence tolerance $\tau = 10^{-10}$. Results from BGC iteration, adaptive quadrature, and prescient quadrature are shown with crosses ($\times$), circles ($\circ$), and plus signs (+) respectively.

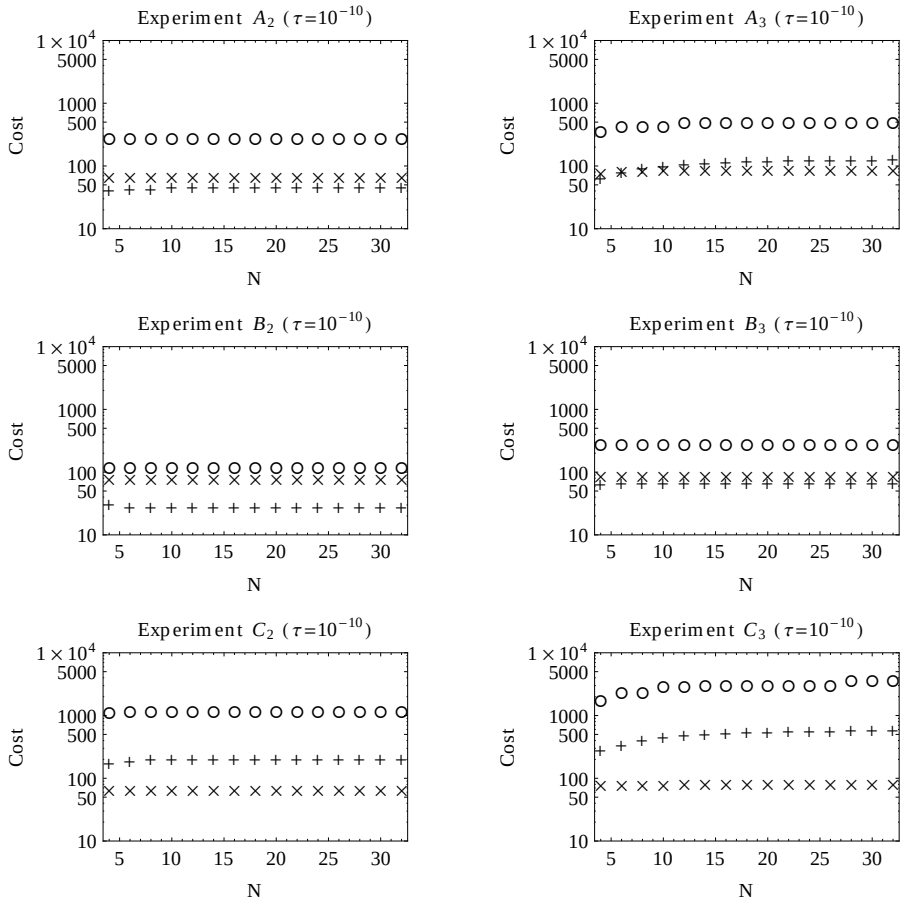


FIGURE 4.11. Cost results from experiment set ABC, intervals 2 and 3, convergence tolerance $\tau = 10^{-10}$. Results from BGC iteration, adaptive quadrature, and prescient quadrature are shown with crosses ($\times$), circles ($\circ$), and plus signs (+) respectively.

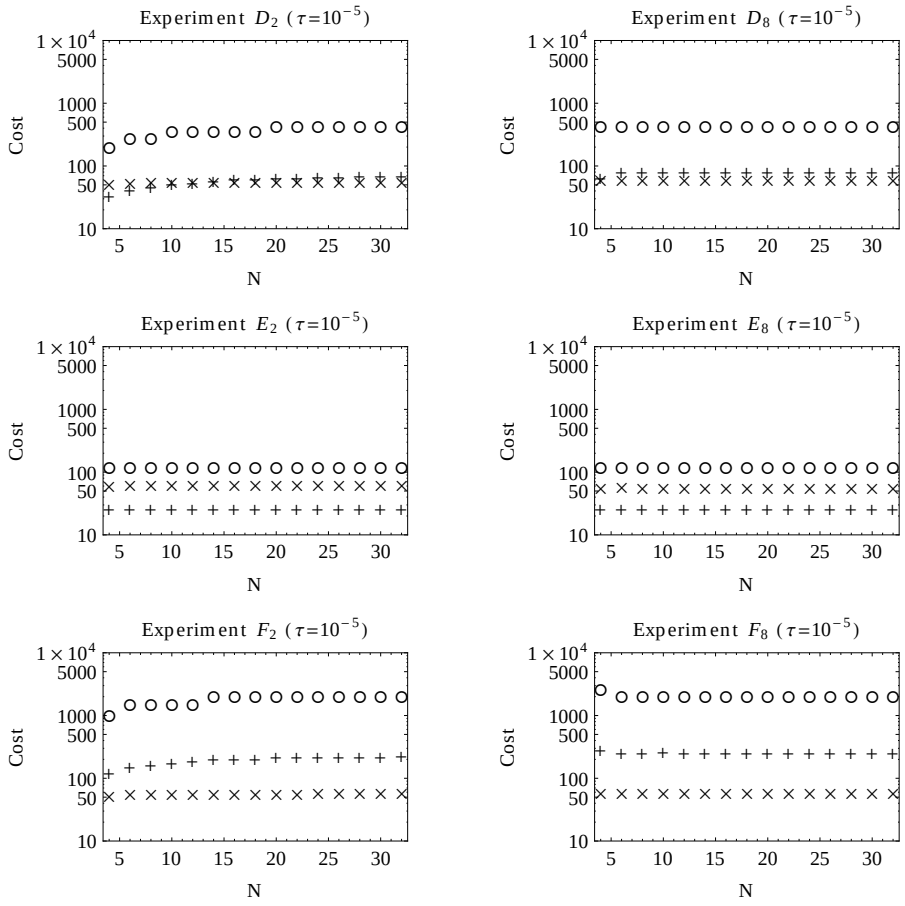


FIGURE 4.12. Cost results from experiment set DEF, $p = 2$ and 8, convergence tolerance $\tau = 10^{-5}$. Results from BGC iteration, adaptive quadrature, and prescient quadrature are shown with crosses ($\times$), circles ($\circ$), and plus signs (+) respectively.

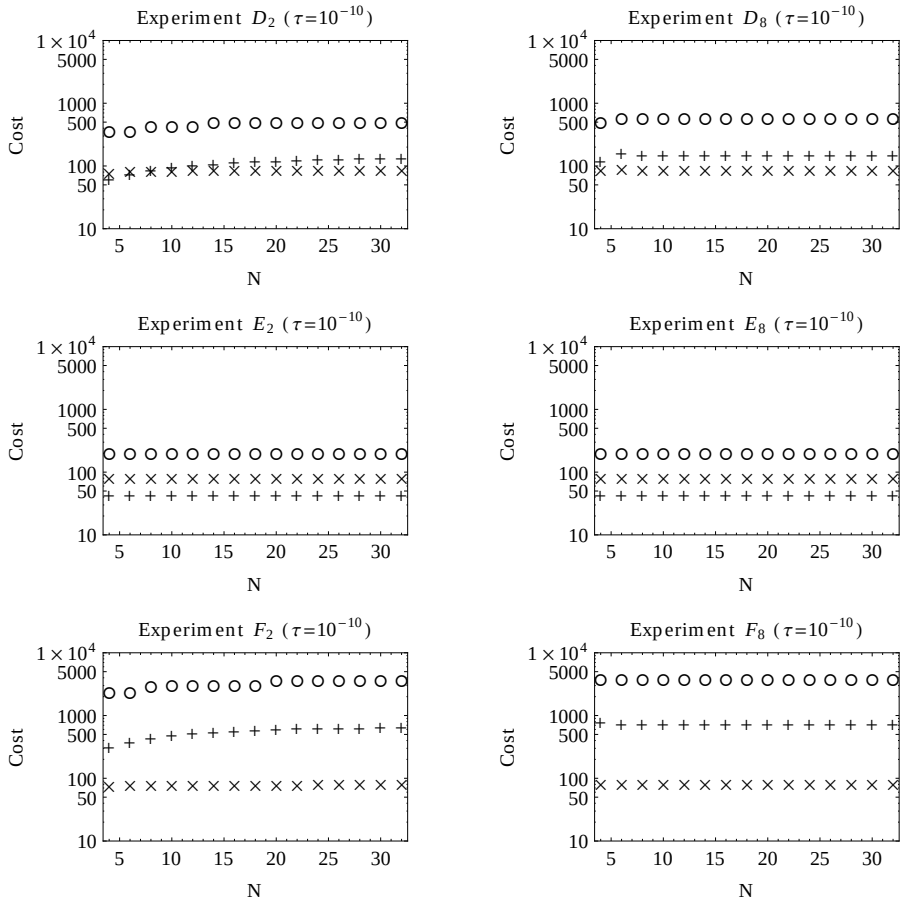


FIGURE 4.13. Cost results from experiment set DEF, $p = 2$ and 8, convergence tolerance $\tau = 10^{-10}$. Results from BGC iteration, adaptive quadrature, and prescient quadrature are shown with crosses ($\times$), circles ($\circ$), and plus signs (+) respectively.