On evaluating multivariate Taylor polynomials

Andreas Griewank†‡, Jan Riehme‡, Jean Utke*
† Matheon Research Center Berlin
‡ Humboldt Universität zu Berlin,
* Argonne National Laboratory, USA

Matheon Workshop in honor of the 60th birthday of Richard Brent
Computing by the Numbers: Algorithms, Precision, and Complexity
July 20 - 21, 2006, Berlin
Table of Contents

- Success Stories in High Order Differentiation.
- Multi-univariate ↔ Multivariate ↔ Uni-univariate.
- Fast Series Manipulation a la Brent et al.
- Interval tightening by Newton and Meanvalue.
- Numerical Experiments and Tentative Conclusions.
Table of Contents

- Success Stories in High Order Differentiation.
- Multi-univariate $\leftrightarrow$ Multivariate $\leftrightarrow$ Uni-univariate.
- Fast Series Manipulation a la Brent et al.
- Interval tightening by Newton and Meanvalue.
- Numerical Experiments and Tentative Conclusions.
Table of Contents

- Success Stories in High Order Differentiation.
- Multi-univariate $\leftrightarrow$ Multivariate $\leftrightarrow$ Uni-univariate.
- Fast Series Manipulation a la Brent et al.
- Interval tightening by Newton and Meanvalue.
- Numerical Experiments and Tentative Conclusions.
Table of Contents

- Success Stories in High Order Differentiation.
- Multi-univariate $\leftrightarrow$ Multivariate $\leftrightarrow$ Uni-univariate.
- Fast Series Manipulation a la Brent et al.
- Interval tightening by Newton and Meanvalue.
- Numerical Experiments and Tentative Conclusions.
Table of Contents

- Success Stories in High Order Differentiation.
- Multi-univariate $\leftrightarrow$ Multivariate $\leftrightarrow$ Uni-univariate.
- Fast Series Manipulation a la Brent et al.
- Interval tightening by Newton and Meanvalue.
- Numerical Experiments and Tentative Conclusions.
Numerical System Inversion

Physical model of a manipulator:

\[ d_1 \quad l_1 \quad \varphi_1 \quad \varphi_2 \quad l_2 \]

- \( d_1 \) is the distance from the winch to the beam.
- \( l_1 \) is the length of the beam.
- \( \varphi_1 \) is the angle between the beam and the rope.
- \( \varphi_2 \) is the angle between the rope and the rope's attachment point.
- \( l_2 \) is the length of the rope.

- The winch is at point \( O \).
- The beam is at point \( A \).
- The rope is attached to point \( S_1 \) and \( S_2 \).
Goal: Compute input for prescribed output!

\[ \tilde{y}_1(t) = \sin \alpha(t) \]
\[ \tilde{y}_2(t) = \cos \alpha(t) \]
ADOL-C vs. Mathematica

C code generated by Mathematica (CForm):

<table>
<thead>
<tr>
<th>derivative order $d$</th>
<th>lines of C code ( y_1^{(d)} )</th>
<th>lines of C code ( y_2^{(d)} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>49</td>
<td>49</td>
</tr>
<tr>
<td>3</td>
<td>416</td>
<td>421</td>
</tr>
<tr>
<td>4</td>
<td>4149</td>
<td>4164</td>
</tr>
<tr>
<td>5</td>
<td>56619</td>
<td>57027</td>
</tr>
</tbody>
</table>
Runtimes:

Taylor coefficients and Jacobians by ADOL-C

Taylor coefficients $y_0, \ldots, y_d$ by ADOL-C

Computation of $y^{(d)}$ (by CForm)
Virtual Machine Tool

\[ F : \mathbb{R}^3 \times \mathbb{R}^{n_p} \rightarrow \mathbb{R}^3 : (w, p) \mapsto x = F(w, p) \]
Evaluation of Restricted Higher Derivative Tensors

\[ \nabla^k_S F (u) \equiv \left. \frac{\partial^k}{\partial v^k} F (u + Sv) \right|_{v=0} \in \mathbb{R}^{3 \times q^k}, \quad u \equiv (w, p) \in \mathbb{R}^{3+n_p}, \quad S \in \mathbb{R}^{(3+n_p) \times q} \]

<table>
<thead>
<tr>
<th>( q )</th>
<th>Maple</th>
<th>ADOL-C</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.12 ms</td>
<td>0.20 ms</td>
<td>0.60</td>
</tr>
<tr>
<td>2</td>
<td>0.13 ms</td>
<td>0.45 ms</td>
<td>0.29</td>
</tr>
<tr>
<td>3</td>
<td>0.14 ms</td>
<td>0.99 ms</td>
<td>0.14</td>
</tr>
</tbody>
</table>

**Table 1: \( n_p = 0 \)**

<table>
<thead>
<tr>
<th>( q )</th>
<th>Maple</th>
<th>ADOL-C</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>0.28 ms</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>0.67 ms</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>1.49 ms</td>
<td></td>
</tr>
</tbody>
</table>

**Table 3: \( n_p = 12 \)**

<table>
<thead>
<tr>
<th>( q )</th>
<th>Maple</th>
<th>ADOL-C</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.02 ms</td>
<td>0.25 ms</td>
<td>4.1</td>
</tr>
<tr>
<td>2</td>
<td>1.20 ms</td>
<td>0.60 ms</td>
<td>2.0</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>1.28 ms</td>
<td></td>
</tr>
</tbody>
</table>

**Table 2: \( n_p = 6 \)**

<table>
<thead>
<tr>
<th>( q )</th>
<th>Maple</th>
<th>ADOL-C</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.02 ms</td>
<td>0.25 ms</td>
<td>4.1</td>
</tr>
<tr>
<td>2</td>
<td>1.20 ms</td>
<td>0.60 ms</td>
<td>2.0</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>1.28 ms</td>
<td></td>
</tr>
</tbody>
</table>

**Table 4: \( n_p = 18 \)**
Taylor Coefficients and Jacobians

Suppose $F$ is $d$ times continuously differentiable on some neighborhood of a point $x_0 \in \mathbb{R}^n$. Then we have

$$y(t) = F \left( \sum_{i=0}^{d-1} x_i t^i \right) = \sum_{i=0}^{d-1} y_i t^i + o(t^{d-1})$$

and $y_i = F_i(x_0, \ldots, x_i)$ has the Jacobian

$$\frac{\partial y_j}{\partial x_i} = \frac{\partial F_j}{\partial x_i} \equiv A_{j-i}(x_0, \ldots, x_{j-i}) \quad \text{for all} \quad 0 \leq i \leq j < d,$$

with $A_i \equiv F'_i(x_0, \ldots, x_i)$ the $i$th Taylor coefficient of the Jacobian of $F$ at $x(t)$, i.e.

$$F'(x(t)) = \sum_{i=0}^{d-1} A_i t^i + o(t^{d-1})$$.
Cartesian Derivative Structure for $n = 2$

\[ \frac{\partial^{i+j} F}{\partial x^i \partial y^j} \text{ needed } \iff (i, j) \in G \supset \mathbb{N} \times \mathbb{N} \]

Practical Worries

- Storage $\iff$ Access Pattern
- Number of Arithmetic Operations

\[ 0 \leq (\tilde{i}, \tilde{j}) \leq (i, j) \in G \implies (\tilde{i}, \tilde{j}) \in G \]
Conversion to Family of univariate Taylor Polynomials

Computed and desired values for $p = 2$ and $d = 4$
Proposition: 
Taylor to Tensor Conversion

Let \( F : \mathbb{R}^n \rightarrow \mathbb{R}^m \) be at least \( d \)-times continuously differentiable at some point \( x \in \mathbb{R}^n \) and denote by \( F_r(x, s) \) the \( r \)th Taylor coefficient of the curve \( F(x + ts) \) at \( t = 0 \) for some direction \( s \in \mathbb{R}^n \). Then we have for any seed matrix \( S = [s_j]_{j=1}^p \in \mathbb{R}^{n \times p} \) and any multi-index \( \mathbf{i} \in \mathbb{N}^p \) with \( |\mathbf{i}| \leq d \) the identity

\[
\frac{\partial^{\mathbf{i}}}{\partial z_1^{i_1} \partial z_2^{i_2} \cdots \partial z_p^{i_p}} F(x + z_1 s_1 + z_2 s_2 + \cdots + z_p s_p) \bigg|_{z=0} = \sum_{|\mathbf{j}|=d} \gamma_{\mathbf{i}\mathbf{j}} F_{|\mathbf{i}|}(x, S \mathbf{j}) ,
\]

where the constant coefficients \( \gamma_{\mathbf{i}\mathbf{j}} \) are given by the finite sums

\[
\gamma_{\mathbf{i}\mathbf{j}} \equiv \sum_{0 < k \leq \mathbf{i}} (-1)^{|\mathbf{i} - \mathbf{k}|} \binom{\mathbf{k}}{\mathbf{j}} \left( \frac{d\mathbf{k}/|\mathbf{k}|}{\mathbf{j}} \right) \left( \frac{|\mathbf{k}|}{d} \right)^{|\mathbf{i}|}.
\]
**Multivariate** ↔ **Uni–univariate**

**Example**

\[ F(x, y) = \sum_{i \geq 0 \leq j} c_{ij} x^i y^j \]

**Substitution**

\[ z(t) = F(t^2, t^3) = \sum_{i \geq 0 \leq j} c_{ij} t^{2i+3j} \]

\[ = c_{00} + c_{10} t^2 + c_{01} t^3 + c_{20} t^4 + c_{11} t^5 + ? t^6 + c_{21} t^7 + O(t^6) \]

⇒ **First 8 univariate coefficients of** \( z(t) \)

yield 6 multivariate coefficients of \( f(x, y) \)
Generally by Chinese Remainder

\[ d \equiv \text{maximal degree of differentiation} \]
\[ n \equiv \text{number of independent variables} \]

\[ d < p_j \quad \text{for} \quad 1 \leq j \leq n, \quad \gcd(p_i, p_j) = 1 \]

\[ m \equiv p_1 \cdot p_2 \ldots p_n \geq d^n, \quad m_i \equiv m/p_i = \prod_{j \neq i} p_j \]

Univariate polynomial

\[ x(t) \equiv (t^{m_1}, t^{m_2}, \ldots, t^{m_n}) : \mathbb{R} \to \mathbb{R}^n \]

with

\[ F(x(t)) = \sum_{j=0}^{m} y_j t^j + o(t^{nm}) \]

identifies all derivatives

\[ \frac{\partial^{(i)} F}{\partial x_1^{i_1} \ldots \partial x_n^{i_n}} \quad \text{with} \quad \sum_{k=1}^{n} i_k \leq d \]

Memory is increased by roughly \( n! \) if \( d \gg n \).
Computational Complexity of Multiplication

Truncated multiplication of one pair of \( n \)-variate polynomials of degree \( d \) involves

\[
\binom{2n + d}{n} \approx \frac{d^{2n}}{(2n)!} \quad \text{if} \quad n \ll d
\]

real multiplications. Other nonlinear elementaries incur similar costs. Multiplication of family of pairs of univariate polynomials of maximal degree \( d \) costs

\[
\binom{n + d - 1}{n} \binom{d + 2}{2} \approx \frac{d^{n+1}}{2 \ n!} \quad \text{if} \quad n \ll d
\]

Multiplication of one pair of univariate polynomials of maximal degree \( m \approx d^n \)

\[
0.5d^{2n} \quad \text{or} \quad c \ d^n \ n \log^2(d)
\]

when basic or fast convolution methods are used, respectively.

On closer expection one finds that the single univariate approach is numerically equivalent to the multivariate approach (almost).
Real analytic $\varphi : \mathbb{R} \rightarrow \mathbb{R}$ has extension

$$\varphi \left( \sum u_i t^i \right) = \sum_{k=1}^{d} \frac{1}{k!} \varphi^{(k)}(u_0) \left( \sum_{i=1}^{d-1} u_i t^i \right)^k$$

Linear ODE on reals

$$\varphi'(u_0) = g(u_0) + m(u_0) \cdot \varphi(u_0)$$

allows recursive evaluation of $v = \varphi(u) = \sum v_i \in \mathcal{R}$

$$v_i = \frac{1}{i} \sum_{j=0}^{i-1} a_j \cdot \tilde{u}_{i-j} \quad \text{with} \quad a_i = g_i + \sum_{j=0}^{i} m_j \cdot v_{i-j}$$

where $\tilde{u}_k \equiv k \cdot u_k$.

Cost $\approx 2$ Convolutions
Most Important Cases

Reciprocal:

\[ v_i = -v_0 \left( \sum_{j=0}^{i-1} v_j u_{i-j} \right) \quad \text{for} \quad i = 1 \ldots d \]

Logarithm:

\[ v_i = \frac{1}{u_0} \left( u_i - \sum_{j=1}^{i-1} \frac{j}{i} v_j u_{i-j} \right) \quad \text{for} \quad i = 1 \ldots d \]

Exponential:

\[ v_i = \frac{1}{i} \sum_{j=0}^{i-1} v_j \tilde{u}_{i-j} \quad \text{with} \quad \tilde{u}_j = j u_j \quad \text{for} \quad i = 1 \ldots d \]

Power:

\[ v_i = \frac{1}{i u_0} \sum_{j=0}^{i-1} v_j u_{i-j} \left[ c - j(c + 1)/i \right] \quad \text{for} \quad i = 1 \ldots d \]
Numerical Example

\[ f = \prod_{i=1}^{n} \sin(x_i) \]

For \( n = 3 \) and \( d = 3 \), we choose \((p_1, p_2, p_3) = (4, 5, 7)\) and so that \( m = 140 = 4 \times 5 \times 7 \) and get as result for \( x = (1.3, 1.6, 1.9) \)

<table>
<thead>
<tr>
<th>index</th>
<th>location</th>
<th>entry</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0,0,3)</td>
<td>60</td>
<td>0.311375</td>
</tr>
<tr>
<td>(0,1,2)</td>
<td>68</td>
<td>0.0266246</td>
</tr>
<tr>
<td>(0,2,1)</td>
<td>76</td>
<td>0.311375</td>
</tr>
<tr>
<td>(0,3,0)</td>
<td>84</td>
<td>0.0266246</td>
</tr>
<tr>
<td>(1,0,2)</td>
<td>75</td>
<td>-0.253026</td>
</tr>
<tr>
<td>(1,1,1)</td>
<td>83</td>
<td>0.00252516</td>
</tr>
<tr>
<td>(1,2,0)</td>
<td>91</td>
<td>-0.253026</td>
</tr>
<tr>
<td>(2,0,1)</td>
<td>90</td>
<td>0.311375</td>
</tr>
<tr>
<td>(2,1,0)</td>
<td>98</td>
<td>0.0266246</td>
</tr>
<tr>
<td>(3,0,0)</td>
<td>105</td>
<td>-0.253026</td>
</tr>
</tbody>
</table>

Maximum relative error vs the directly computed entries 4.38778e-16.

Alternatively we propagate 10 univariate directions of degree 3 and get the same result this time with maximum relative error of 1.83766e-14.
Fast Evaluations á la Brent et al.

Reciprocal:

\[ v = 1/u \equiv \text{rec}(u) \quad \text{solves} \quad f(v) \equiv 1/v - u = 0 \quad \text{yielding by Newton} \]

\[ N(v) = v + (1/v - u) \times v \times v = v + (1 - u \times v) \times v \]
Fast Evaluations á la Brent et al.

Reciprocal:

\[ v = 1/u \equiv \text{rec}(u) \] solves \[ f(v) \equiv 1/v - u = 0 \] yielding by Newton

\[ N(v) = v + (1/v - u) \ast v \ast v = v + (1 - u \ast v) \ast v \]

Logarithm:

\[ v = \log(u) \] satisfies \[ \dot{v} = \dot{u} \ast (1/u) \] where \[ \dot{w}_j = (j + 1)w_{j+1} \] for \[ w = u, v \]
Fast Evaluations à la Brent et al.

Reciprocal:

\[ v = 1/u \equiv \text{rec}(u) \] solves \[ f(v) \equiv 1/v - u = 0 \] yielding by Newton

\[ N(v) = v + (1/v - u) \cdot v \cdot v = v + (1 - u \cdot v) \cdot v \]

Logarithm:

\[ v = \log(u) \] satisfies \[ \dot{v} = u \cdot (1/u) \] where \[ \dot{w}_j = (j + 1)w_{j+1} \] for \[ w = u, v \]

Exponential:

\[ v = \exp(u) \] solves \[ f(u) \equiv \log(v) - u = 0 \] yielding by Newton

\[ N(v) = v \cdot (1 + u - \log v) \]
Fast Evaluations á la Brent et al.

Reciprocal:

\[ v = 1/u \equiv \text{rec}(u) \text{ solves } f(v) \equiv 1/v - u = 0 \text{ yielding by Newton} \]

\[ N(v) = v + (1/v - u) \cdot v \cdot v = v + (1 - u \cdot v) \cdot v \]

Logarithm:

\[ v = \log(u) \text{ satisfies } \dot{v} = \dot{u} \cdot (1/u) \text{ where } \dot{w}_j = (j + 1)w_{j+1} \text{ for } w = u, v \]

Exponential:

\[ v = \exp(u) \text{ solves } f(u) \equiv \log(v) - u = 0 \text{ yielding by Newton} \]

\[ N(v) = v \cdot (1 + u - \log v) \]

\[ \text{Quadratic convergence } = \text{ doubling of correct terms} \]

\[ \implies \text{ Complexity } \sim \text{ Cost (Workhorse Convolution)} \]
Mean Value Form in Taylor Calculus

\[ \varphi(\bar{u} + \Delta u) \approx \varphi(\bar{u}) + \varphi'(\bar{u}) \ast \Delta u \]

where

\[ \mathcal{R} \ni \varphi'(\bar{u}) \Rightarrow \text{Töplitz matrix} \in \mathbb{R}^{d \times d} \]

corresponds to pre-accumulation of local Jacobian.

Mean value from

\[ v = \varphi(u) \subset \varphi(\bar{u}) + \varphi'(u) \ast (u - \bar{u}) \]

with interval radius

\[ \| \varphi'(\bar{u}) \|_1 \rho(u) \approx \rho(v) \leq \| \varphi'(u) \|_1 \rho(u) \].
Codependence and Postiteration

While naive convolution is single usage and thus optimal

Reciprocal:

\[ v = 1/u \quad \text{implies} \quad v_2 = -v_0(v_0u_2 + v_1u_1) = (u_1^2 - (v_0u_2)/u_0^3) \]
Codependence and Postiteration

While naive convolution is single usage and thus optimal

Reciprocal:

\[ v = 1/u \quad \text{implies} \quad v_2 = -v_0(v_0u_2 + v_1u_1) = \frac{u_1^2 - (v_0u_2)}{u_0^3} \]

Meanvalue:

\[ M(v) = 1/\ddot{u} - v \ast v \ast (u - \ddot{u}) \]
Codependence and Postiteration

While naive convolution is single usage and thus optimal

**Reciprocal:**

\[ v = 1/u \quad \text{implies} \quad v_2 = -v_0(v_0u_2 + v_1u_1) = \frac{u_1^2 - (v_0u_2)}{u_0^3} \]

**Meanvalue:**

\[ M(v) = 1/\ddot{u} - v * v * (u - \ddot{u}) \]

**Newton:**

\[ N(v) = \dddot{v} + (1 - u * \dot{v}) * \ddot{v} + 2(1 - \ddot{u} * v)(v - \dot{v}) \]
Codependence and Postiteration

While naive convolution is single usage and thus optimal

Reciprocal:

\[ v = 1/u \quad \text{implies} \quad v_2 = -v_0(v_0u_2 + v_1u_1) = (u_1^2 - (v_0u_2)/u_0^3 \]

Meanvalue:

\[ M(v) = 1/\ddot{u} - v \ast v \ast (u - \ddot{u}) \]

Newton:

\[ N(v) = \ddot{v} + (1 - u \ast \ddot{v}) \ast \ddot{v} + 2(1 - \ddot{u} \ast v)(v - \ddot{v}) \]

Initialization

\[ I(u) \quad \text{based on classical recurrence.} \]
General Procedure

\[
v := I(u); \quad v := v \cap M(v) \\
v := v \cap N(v) \quad \text{until} \quad v \subseteq N(v)
\]

Other cases:
\[
v = \log(u) \quad \implies
I(u) = \text{lifted} \quad \dot{v} = \text{rec}(u)
M(v) = \log(\tilde{u}) + \text{rec}(u) \ast (u - \tilde{u})
N(v) = \tilde{v} + (u \ast \exp(-\tilde{v}) - 1) + (1 - \tilde{u} \ast \exp(-v)) \ast (u - \tilde{u})
\]

\[
v = \exp(u) \quad \implies
I(u) = \text{classical recurrence}
M(v) = \exp(\tilde{u}) + v \ast (u - \tilde{u})
N(v) = \tilde{v} + (u - \log(\tilde{v})) \ast \dot{v} + (\tilde{u} - \log(v)) \ast (u - \tilde{u})
\]
Reciprocal, \( d=25, \varepsilon=0.001, T_i = [-\varepsilon, \varepsilon] + (-1)^i / (2+i^{1.01}) \)
Reciprocal, $d=25$, $\varepsilon=0.001$, $T_i = [-\varepsilon, \varepsilon] + (-1)^i / \sqrt{1 + i}$
Logarithm, $d=50$, $\varepsilon=0.001$, $T_i = [-\varepsilon, \varepsilon] + (-1)^i / (2+i^{1.01})$

- Recurrence
- Mean Value Form
- Integral form with reciprocal via Newton (19)
- Newton iteration (14)
- Lower bound
Logarithm, $d=50$, $\varepsilon=0.001$, $T_i = [-\varepsilon,\varepsilon] + (-1)^i / \sqrt{1+i}$
Exponential, \( d=100, \varepsilon=0.001, T_i = [-\varepsilon, \varepsilon] + (-1)^i / (2 + i^{1.01}) \)}
Exponential, \( d=100, \varepsilon=0.001, T_i = [-\varepsilon, \varepsilon] + (-1)^i / \sqrt{1 + i} \)

- Recurrence
- Mean Value Form
- Interative Mean Value Form
- Newton iteration (0), init via recurrence, log via recurrence
- Newton iteration (0), init via recurrence, log via Integral
- Lower bound
Power, $d=50$, $\varepsilon=0.001$, $c=0.4$, $T_i = [-\varepsilon,\varepsilon] + (-1)^i / (2 + i^{0.01})$
Power, $d=50$, $\varepsilon=0.001$, $c=0.4$, $T_i = [-\varepsilon, \varepsilon] + (-1)^i / (2 + i^{1.01})$
Concluding Remarks and Questions

- Conversion: multivariate $\rightarrow$ univariate family efficient.

- Conversion: multivariate $\rightarrow$ single univariate dubious.

- Conflict between fast and thight interval evaluation ????.

- Interval result can/must be tightened by postiterations.
Concluding Remarks and Questions

- Conversion: multivariate $\rightarrow$ univariate family efficient.

- Conversion: multivariate $\rightarrow$ single univariate dubious.

- Conflict between fast and thight interval evaluation ????.

- Interval result can/must be tightened by postiterations.
Concluding Remarks and Questions

- Conversion: multivariate $\rightarrow$ univariate family efficient.

- Conversion: multivariate $\rightarrow$ single univariate dubious.

- Conflict between fast and thight interval evaluation ????.

- Interval result can/must be tightened by postiterations.
Concluding Remarks and Questions

- Conversion: multivariate $\rightarrow$ univariate family efficient.

- Conversion: multivariate $\rightarrow$ single univariate dubious.

- Conflict between fast and thight interval evaluation $????$.

- Interval result can/must be tightened by postiterations.