Tangent Linear Models by Operator Overloading

No Pain: Some Gain

Formerly: Algorithmic Differentiation for Analysis of Global Change and the Carbon Cycle

Ian G. Enting
MASCOS
The University of Melbourne
Acknowledgments

- The Center of Excellence for Mathematics and Statistics of Complex Systems (MASCOS) is funded by the Australian Research Council (ARC).
- My fellowship at MASCOS is supported by CSIRO through a sponsorship agreement.
- The Fortran-90 development is supported by the ARC Earth System Science Network (ARCNESS).
- Collaborators: Cathy Trudinger and YingPing Wang of CSIRO Marine and Atmospheric Research and members of the MATCH working group on the Brazilian Proposal.
Summary

- Algorithmic differentiation (AD) in analysing models
- Implementing tangent linear models using operator overloading in C++ and Fortran-90/95
- Applications of AD in:
  - Analysing the Brazilian proposal
  - Calibrating CASACNP terrestrial carbon model (with N and P)

Progress report

MSRI workshop on Data Assimilation for the Carbon Cycle
Model Analysis

Most common model calculation is forward projection by (numerical) integration of DEs. Much model analysis involves differentiation:

- **Initialisation** Solving for zero rate of change
- **Sensitivity analysis** — derivatives with respect to parameters
- **Calibration** — techniques such as Maximum Likelihood imply optimisations, facilitated by use of derivatives
- **Data assimilation** — real-time model adjustment — dynamic calibration
Tangent Linear Model (TLM)

For $N$ DEs: \[ \frac{d}{dt} x_j = g_j(\{x_k\}, \alpha, t) \] for $j = 1, N$

we can define sensitivities as

\[ y_j = \frac{\partial}{\partial \alpha} x_j \] for $j = 1, N$ or \[ y_{j,p} = \frac{\partial}{\partial \alpha_p} x_j \]

to give ‘tangent linear model(s)’:

\[ \frac{d}{dt} y_m = \frac{\partial}{\partial \alpha} g_m(\{x_k\}, \alpha, t) + \sum_n \frac{\partial}{\partial x_n} g_m(\{x_k\}, \alpha, t) y_n \]

\[ \frac{d}{dt} y_{m,p} = \frac{\partial}{\partial \alpha_p} g_m(\{x_k\}, \alpha, t) + \sum_n \frac{\partial}{\partial x_n} g_m(\{x_k\}, \alpha, t) y_{n,p} \]
Using the TLM

The Tangent Linear Model (TLM) gives sensitivities with respect to parameters — useful for analysis of uncertainty. Also provides a linearisation of a model relation defined in terms of DEs. Useful for variational calibration

\[ \Theta = [z_j - \mathcal{H}(\alpha)]^T R^{-1} [z_j - \mathcal{H}(\alpha)] - \ln[\text{Pr}_{\text{prior}}(\alpha)] \]

‘model relation’ \( \mathcal{H}(\alpha) \) from solution of DEs.

\[ d\Theta = 2[z_j - \mathcal{H}(\alpha)]^T R^{-1} H d\alpha - \ldots \]

The TLM gives the linearisation \( H \)
Algorithmic Differentiation (AD)

Differentiation by successive use of chain rule. For binary operation $c = f(a, b)$,

$$\frac{\partial c}{\partial \alpha} = \frac{\partial f}{\partial a} \cdot \frac{\partial a}{\partial \alpha} + \frac{\partial f}{\partial b} \cdot \frac{\partial b}{\partial \alpha}$$

e.g.

$$c = a + b \quad \rightarrow \quad \frac{\partial c}{\partial \alpha} = \frac{\partial a}{\partial \alpha} + \frac{\partial b}{\partial \alpha}$$

$$c = a \times b \quad \rightarrow \quad \frac{\partial c}{\partial \alpha} = b \times \frac{\partial a}{\partial \alpha} + a \times \frac{\partial b}{\partial \alpha}$$

Convert program to code for derivatives, one operation at a time.
Computational Complexity of AD

For $u_k \rightarrow x_j(1) \rightarrow \ldots x_j(m) \cdots \rightarrow x_j(M) \rightarrow v_j$,

$m$ is operation count, Jacobian is: $J_{jk} = \frac{\partial v_j}{\partial u_k}$

$$J_{jk} = \sum \frac{\partial v_j}{\partial x_n(M)} \ldots \frac{\partial x_{n'}(m)}{\partial x_{n''}(m-1)} \ldots \frac{\partial x_{p'}(1)}{\partial x_p(0)} \frac{\partial x_p(0)}{\partial u_k}$$

Tangent: $\frac{\partial v_j}{\partial \alpha} = \sum_k J_{jk} \frac{\partial u_k}{\partial \alpha} \quad \text{vector } \times \text{ sparse matrix product}$

Gradient: $\frac{\partial \phi}{\partial u_k} = \sum_j J_{jk} \frac{\partial \phi}{\partial y_j}$

Adjoint model achieves efficiency of vector $\times$ sparse matrix by using chain rule backwards in time.

---

MSRI workshop on Data Assimilation for the Carbon Cycle
Adjoints (of TLM)

\[
\frac{d}{dt} y_{m,p} = \frac{\partial}{\partial \alpha_p} g_m(\{x_k\}, \alpha, t) + \sum_n \frac{\partial}{\partial x_n} g_m(\{x_k\}, \alpha, t) y_{n,p}
\]

\[
\frac{d}{dt} y_{m,p} = f_{m,p}(t) + \sum_n g_{m,n}(t) y_{n,p}
\]

TLM is **linear** and therefore has a (linear) Green’s function operator which has an adjoint. Jacobian is projection of operator onto matrix.

\( J_{k,1} \) gives \( \frac{\partial v_k}{\partial u_1} \): carry forward single set of \( \frac{\partial x_m}{\partial u_1} \)

\( J_{1,j} \) gives \( \frac{\partial v_1}{\partial u_j} \): carry backward single set of \( \frac{\partial v_1}{\partial x_m} \)
Sparse Matrix Factorisation

Convert matrix $\times$ vector to successive multiplications of vector by sparse matrices.

e.g. FFT: $\tilde{f}(k) = \sum_n \exp(2\pi i nk/N) f(n)$ for $N = 2^M$

$$\tilde{f} = M_N f = \prod_{j=1}^M L_j f$$

Jacobian for AD is product of sparse matrices: $J_m$ for $c = a \cdot op. b$ is identity, except for column $c$, rows $a,b,c$.

Applications in lattice statistical mechanics:

.. conventional algorithm would .. run for time comparable to lifetime of the universe on .. fastest supercomputer to achieve .. same results .. Guttmann and Enting,1988, J Phys A, 21 L165

MSRI workshop on Data Assimilation for the Carbon Cycle
Fast Fourier Transform

FFT expresses transform of size \( N = 2^M \) as sparse matrix product

\[
\tilde{f} = M_N f = \prod_{j=1}^{M} L_j f
\]

with \( f \) in ‘bit-reversed’ order,

\[
M_N = \begin{bmatrix}
I & W_1 \\
W_2 & I
\end{bmatrix} \times \begin{bmatrix}
M_{N/2} & 0 \\
0 & M_{N/2}
\end{bmatrix}
\]

where \( W_1 \) and \( W_2 \) are diagonal, \( (\exp(2\pi ik/N)) \)

Thus \( N^2 = 2^{2M} \) multiplications become

\( 2MN = 2^{M+1}M \)
Approaches to AD

Hand-code program to calculate derivatives — laborious, error-prone and must be repeated each time the model changes.

Symbolic algebra (e.g. Mathematica) — problematic for adjoints.

Tangent/adjoint compilers — transform source into code for tangent or adjoint models.

Operator overloading to produce a ‘script’ that is analysed to give code for the derivatives.

Use operator overloading capabilities directly — straightforward for tangent-linear-model, but restricted applicability to adjoint models.
Operator Overloading (C++)

Replace real variable $x$, (type `double`), with composite variable $\tilde{x}$ (type `Xvar`), representing both value $x$ and its derivatives with respect to $K$ model quantities, $\alpha_k$ as:

$$\tilde{x}_0 = x \quad \text{and} \quad \tilde{x}_k = \frac{\partial}{\partial \alpha_k} x \quad \text{for} \quad k = 1, K$$

Operator overloading implements $\tilde{c} = \tilde{a} \ast \tilde{b}$, representing:

$$\tilde{c}_0 = \tilde{a}_0 \ast \tilde{b}_0 \quad \text{and} \quad \tilde{c}_k = \tilde{a}_0 \ast \tilde{b}_k + \tilde{a}_k \ast \tilde{b}_0$$

Overloaded functions, $\tilde{c} = f(\tilde{a})$, represent:

$$\tilde{c}_0 = f(\tilde{a}_0) \quad \text{and} \quad \tilde{c}_k = f'(\tilde{a}_0) \ast \tilde{a}_k$$

where $f'(.)$ denotes the derivative of $f(.)$
Class Definitions

Fragment of C++ class definition to implement operator overloading:

class Xvar{
public:
static const int ns = _NUMDERIVS+1;
double xs[_NUMDERIVS+1];
Xvar operator*(Xvar);
...
};

Xvar Xvar::operator*(Xvar b){
Xvar c;
for (int i=1; i < ns; i++)
    c.xs[i] = xs[i]*b.xs[0]+xs[0]*b.xs[i];
c.xs[0] = xs[0]*b.xs[0];
return c;}

...
Usage

Original

double F_co2(double c){
double a;
a = log(c/280.0)*5.35;
return a;
}
...
double cc;
...
ff = F_CO2(cc)

Transformed

Xvar F_co2(Xvar c){
Xvar a;
a = log(c/280.0)*5.35;
return a;
}
...
Xvar cc;
// Derivatives wrt
// initial value of cc
cc.set(280,1);
...
ff = F_CO2(cc)

MSRI workshop on Data Assimilation for the Carbon Cycle
Definitions (Fortran-90/95)

type x_var
real, dimension(0:max_d) :: v
end type

interface operator (*)
module procedure xv_mul, xv_mulr, xv_rmul, xv_muli, xv_imul
end interface

...
Putting AD into models

Need to modify model by changing:

- type declarations
- output
- initialisation (and input)
- other surprises ???

Use syntax checking of compiler to help ensure validity.

Real = x_var should be undefined, detection by compiler implies failure to declare all neccessary variables.
## Basic Operator Requirements

A basic set:

### Binary Operators

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(=)</td>
<td>Y</td>
<td>N/A</td>
<td>Y</td>
<td>Y</td>
<td>N/A</td>
</tr>
<tr>
<td>(+)</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>(-)</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>(*)</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>(/)</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
<tr>
<td>(^{\star})</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>Y</td>
<td>–</td>
</tr>
</tbody>
</table>

### Unary Operations (and intrinsics)

<table>
<thead>
<tr>
<th>Op</th>
<th>(-)</th>
<th>sqrt</th>
<th>cos</th>
<th>sin</th>
<th>log</th>
<th>exp</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
<td>Y</td>
</tr>
</tbody>
</table>

---

MSRI workshop on Data Assimilation for the Carbon Cycle
Other Requirements

- Output routines for variables with derivatives — (Fortran write accepts type(x_var) as Fortran array)
- Ability to identify variables for differentiation
- Deal with non-differentiable operations: max, .GT. etc.
- Miscellaneous: error trapping, version identifier, initialise number of derivatives, match variable names to derivatives.
Extensions of overloading

Differentiation —

• Higher derivatives
• Taylor’s series

Other —

• Multi-region (or multi-category in general)
• Isotopes?

C++ implementation of second derivatives used in analysis of Brazilian Proposal.
Brazilian Proposal

Tabled by Brazil during negotiations leading to Kyoto Protocol — Flicked-passed to Subsidiary Body for Scientific and Technical Advice (SBSTA).

Proposes that emission reduction targets should be proportional to nation’s relative responsibility for the greenhouse effect.

Issues:

- Indicator? What quantity is used as a measure of the greenhouse effect?
- For what period of emissions is responsibility attributed?
- How are non-linear responses attributed?
Timescales

$\text{CO}_2$ concentrations and consequent warming, partitioned according to time of emission.

Lowest bands are from pre-1960 emissions, next from 1960 to 1980 emissions, etc. Increase in contribution to warming after time of emissions from ‘committed warming’ effect.

MSRI workshop on Data Assimilation for the Carbon Cycle
Brazilian Proposal as Derivatives

As example, use indicator $T^* = T_{CO2}(2100) =$ warming in 2100 from $CO_2$ emissions. $T^*$ is to be attributed to emissions $E_j(t)$ from country $j$ with $E(t) = \sum_j E_j(t)$.

Differential attribution to country $j$ of emissions at time $t$ is

$$\frac{\partial T^*}{\partial E_j(t)} E_j(t) = \frac{\partial T^*}{\partial E(t)} E_j(t) = S(t) E_j(t)$$

where $S(t)$ is a Frechet derivative.

Cumulated attribution: $T_j^* = \int S(t) E_j(t) \, dt$
Results: Frechet Derivatives

Assumes IS92a emissions. Represents temperature by response function. Linear responses for ocean and biotic carbon, coupled non-linearly to atmospheric CO$_2$ (as in CSIRO study).

\[ \frac{\partial}{\partial E(t)} T(\tau) \text{ for } \tau = 2000, 2050, 2100. \]

Decrease as $t \to \tau$ shows ‘committed warming’. At any time, warming from the most recent releases is yet to happen.

MSRI workshop on Data Assimilation for the Carbon Cycle
Implications

• For a given indicator, $T^*$, calculation of $S(t)$ allows attribution to any nation.

• $S(t)$ most efficiently calculated from adjoint model, but for multiple indicator times, tangent linear model not too too inefficient.

• Sensitivity of $T^*_j$ to model uncertainties can be obtained as second derivatives.

• Sensitivity of $T^*_j$ to uncertainties in emissions can be obtained as

$$\text{Var}[T^*_j] = \int \int S(t) \text{Cov}[E_j(t), E_j(t')] S(t') \, dt' \, dt$$
CASACNP AD Project

- Develop Fortran-90/95 implementation of AD by operator overloading;
- Use CASACNP as test case of AD in existing model;
- Explore use of CASACNP derivatives for calibration;
- Document procedure for ARC Network for Earth System Science.

Development funding obtained from ARCNESS for CASACNP and extension to CABLE.
CASACNP

Extend CASA terrestrial carbon model to include nitrogen and phosphorus. (Wang, Field and Houlton).

- Initially single region, 8 C pools, 9 N pools, 12 P pools.
- Models response to addition of P, N and competition between N-fixers and non-fixers.
- Validation transect (of soil age) in Hawaii.
- Initial conversion to use AD successful: May 2006 — development on-going.

MSRI workshop on Data Assimilation for the Carbon Cycle
Putting AD in CASACNP

• Added $\max$ and $\min$ to initial AD definitions
• Redeclare types of variables — global edit — not efficient code
• Convert input to write to value — initialise derivatives to zero using compiler option
• Identify variables for differentiation
• Explicit re-writes of .GT. tests using values
• Explicit loops for $array = real$ — explicit $1/delt$
Progress on CASACNP

- Successful proof-of concept of AD by operator overloading in Fortran-95.
- Successful runs using ‘brute-force’ conversion of CASACNP (May 2006)
- *Development on-going, supported by ARCNESS*
Future directions

● Incorporate AD into calibration tools.
● Second derivatives in Fortran.
● Explore overloading for Fortran vector operations.
● Develop calibration strategy for CASACNP and other carbon components of Australian Community Climate and Earth System Simulator (ACCESS).
Conclusions

Algorithmic differentiation —
Operator overloading is a straightforward way of developing tangent linear models (and obtaining higher derivatives if needed).

Brazilian Proposal —
Attribution in terms of derivatives is readily calculated using algorithmic differentiation. Higher derivatives give sensitivities.

Global change —
Potential should extend to other analyses of uncertainties in global change.

MSRI workshop on Data Assimilation for the Carbon Cycle
Further Information


MATCH website (Brazilian Proposal): http://www/match-info.net


MSRI workshop on Data Assimilation for the Carbon Cycle