Fortran codes recently differentiated by means of TAF

Ralf Giering and Thomas Kaminski

*Fast*Opt

Copy of presentation at http://FastOpt.com

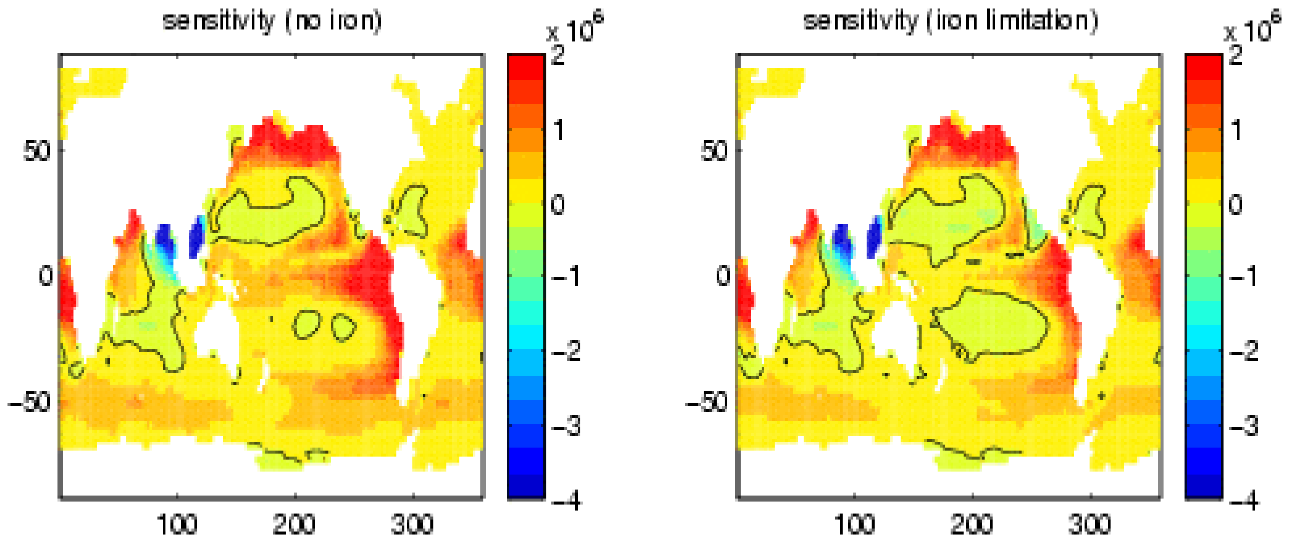
Outline

- Applications
  - ocean/atm. model : MITgcm +biogeochemistry +seaice
  - atmosphere transport model : NIRE-CTM
  - CFD: FLOWer
  - atmosphere model : fvGCM
- Parallelisation (MPI, OpenMP)
- TAF a Fortran-95 source-to-source tool
- Performance
- Summary

AD of biogeochemistry in MITgcm
with MIT (Dutkiewicz, Follows, Heimbach, Marshall)

- AD for tracer code and carbonate chemistry (Dutkiewicz and Follows)
- ~4000 lines of Fortran 77 (without comments) in addition to MITgcm
- Parallelisation: MPI + OpenMP
- Tangent Linear and adjoint generated by TAF
- To be used by MIT for sensitivity studies, parameter estimation, data assimilation ...

dJ/dα
Sensitivity of data fit (phosphate) to max. export rate (Courtesy P. Heimbach)

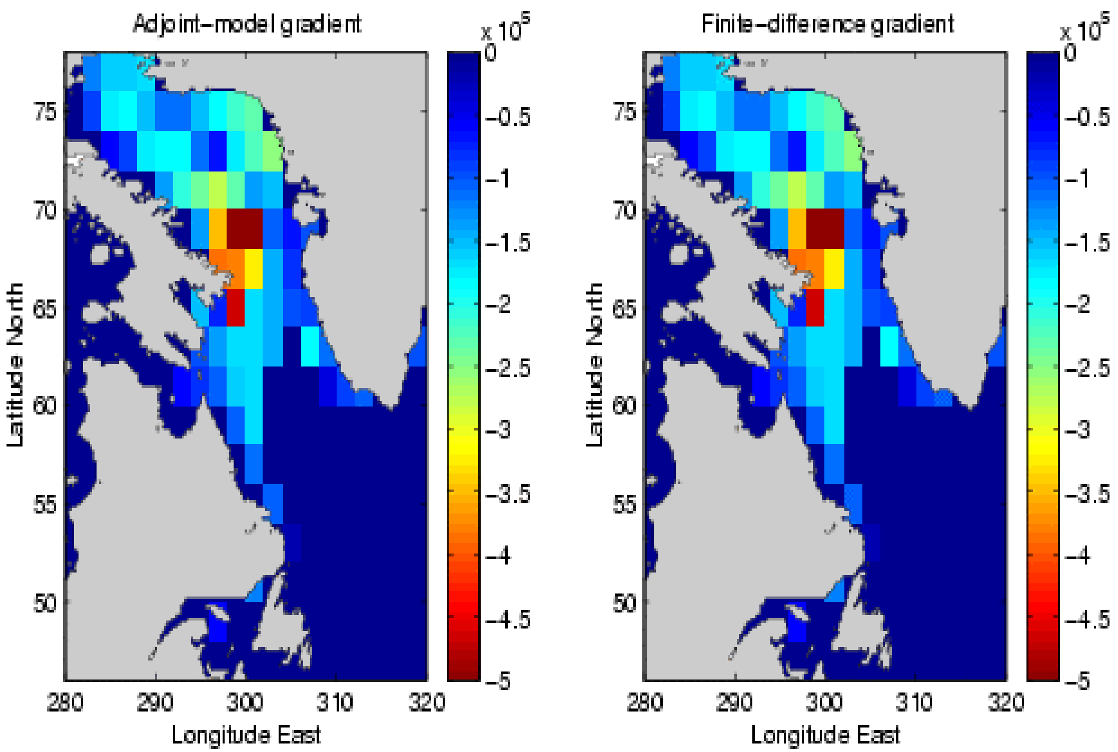


FastOpt

AD of sea-ice in MITgcm
with NASA-JPL-ECCO (Heimbach, Menemenlis, Zhang)

- Sea-ice model based on Hibler (1979 and 1980) and Zhang (1998 and 2000)
- ~3000 lines of Fortran 77 (without comments) in addition to MITgcm
- Parallelisation: MPI + OpenMP
- Tangent Linear and adjoint generated by TAF
- Applications in progress...
- To be used by JPL (ECCO) and Johns Hopkins for
  - Sensitivity studies,
  - Parameter estimation,
  - Data Assimilation ...

first gradient tests
(Courtesy D. Menemenlis)



FastOpt

NIRE-CTM

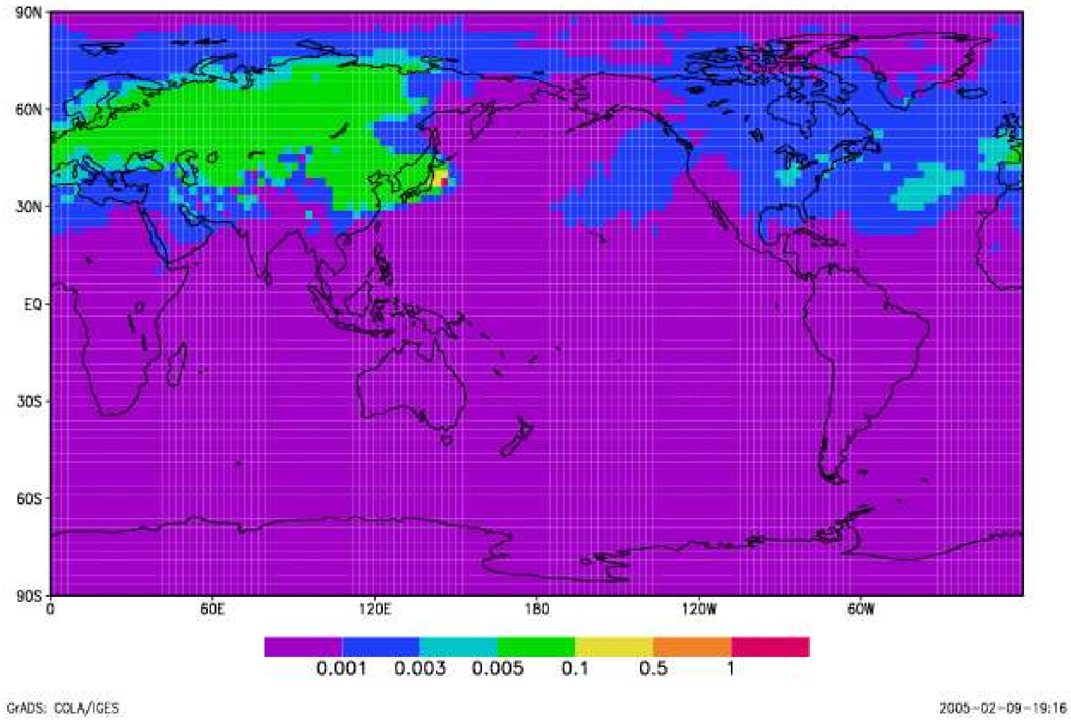
joint project with S. Taguchi (AIST)

NIRE CTM (Taguchi, 1996, JGR):

- atmospheric transport model for passive tracers
- solves continuity equation
- simulates space-time distribution of passive tracers
  from prescribed initial- and boundary (sources and sinks) conditions
- 860 lines of Fortran 77 code
- adjoint needed
  - to provide sensitivity of tracer concentration
    with respect to sources and sinks
  - for assimilation of observed concentration
- adjoint for short integration periods (up to one month, no checkpointing)
- relative performance (multiples of function evaluation):
  - TLM: 1.0
  - ADM 1.5

NIRE-CTM

joint project with S. Taguchi (AIST)



Sensitivity of concentration at Sendai (Japan) to surface sources over seven day period

*Fast*Opt

FLOWer

joint work with B. Eisfeld, N. Gauger, N. Kroll (DLR)

Simple test configuration:

- 2d NACA12
- k-omega (Wilcox) Turbulence
- cell-centred metric
- 2 time steps on fine grid
- d lift/ d alpha

Steps:

- Modificationen of FLOWer code (TAF-directives, small changes etc.)
- tangent-linearer Code (for verification and as intermediate result)
- adjoint code -> fast adjoint code

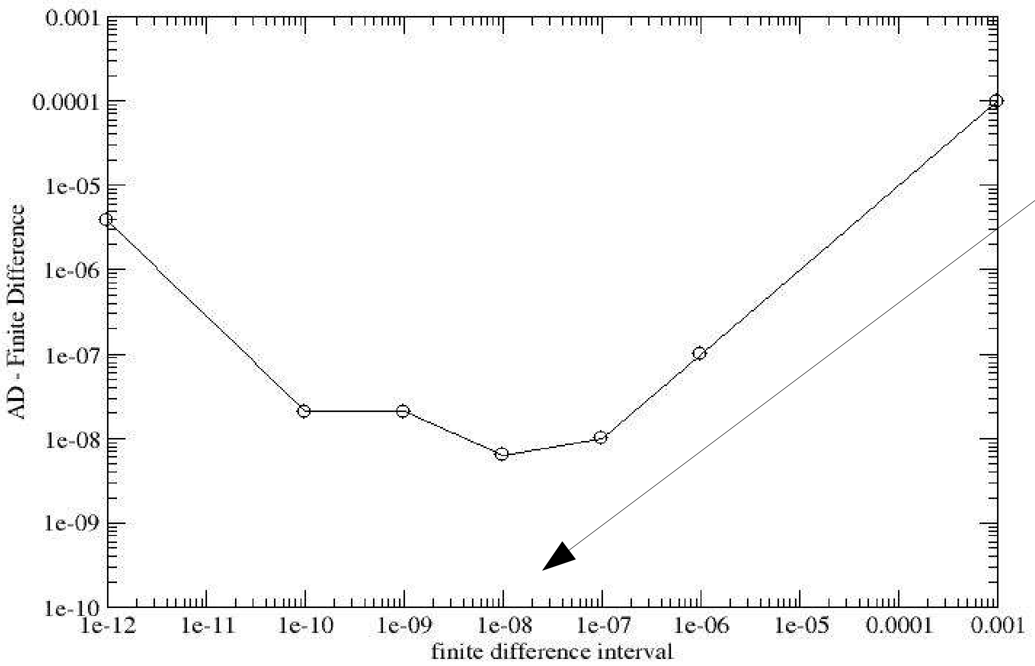
main challenges:

- many goto-statements (error exits)
  -> most goto statements are replaced automatically by sed in preprocess
- dynamic memory management (all fields are stored in one big array)

FLOWer

Verifiction adjoint/tangent linear

```
 **************************************************
 CHECK OF TLM USING eps = 0.100E-07
 **************************************************
     I         x(i)  delta f/eps     grad f   RELATIVE ERR
     1 0.734000E+00 -.304623E+00 -.304623E+00 0.641981E-08
 **************************************************
```

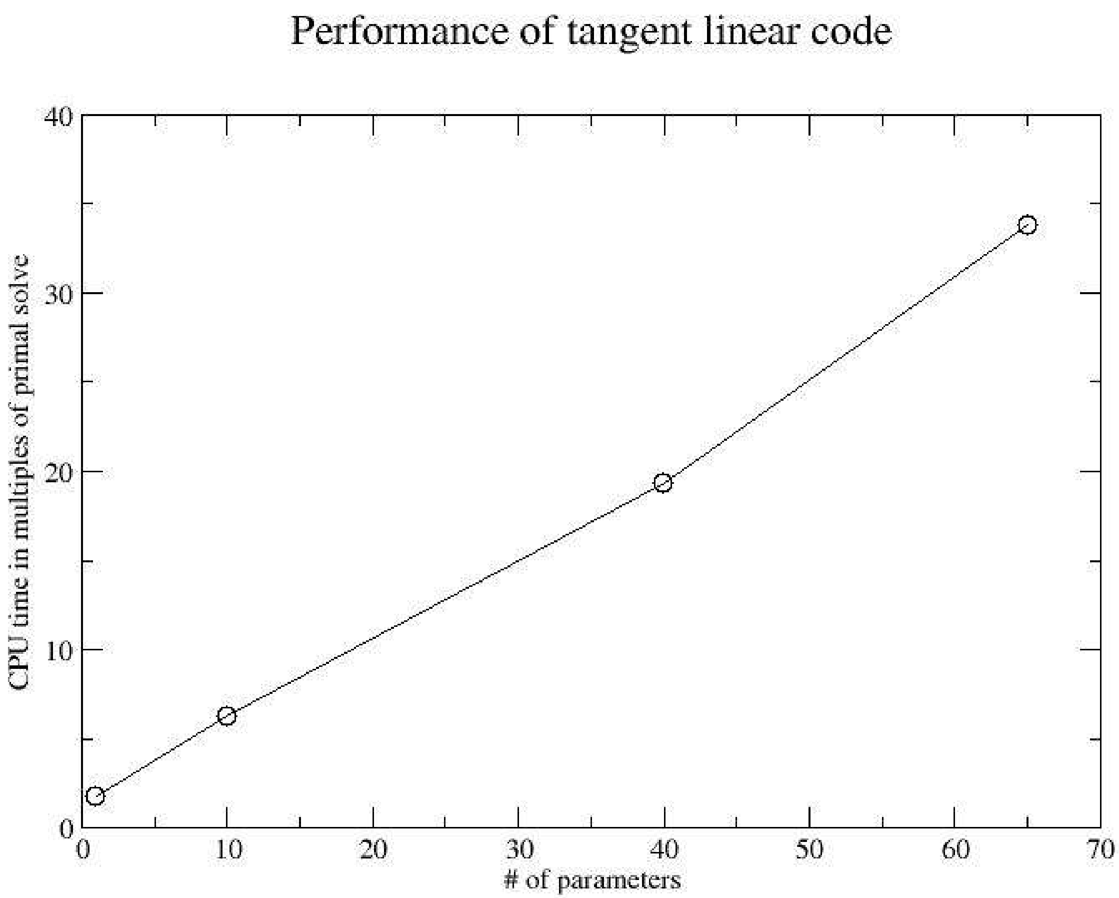


*Fast*Opt

FLOWer

Performance tangent linear

Verhalten einer Konfiguration mit mehreren Paramtern (Designvariablen) simuliert durch gleichzeitige mehrfache Berechnung der Sensitivität bzgl. alpha



Mit Optimierung durch Fortran-Compiler

Status ADFLOWer

done:

- ✔ TLM generated automatically (378 k lines of Fortran)
- ✔ TLM verified in test configuration
- ✔ ADM generated automatically (352 k lines of Fortran)
- ✔ ADM verified in test configuration

in progress:

- Increase performance of ADM
- Reduction of TAF resources to prozess code
  status: TLM ~30 min / ~1.3 GB, ADM ~16 min / ~ 0.7 GB

more:

- multigrid
- parallelisation
- more turbulence models
- sensitivities to design variables

AD of finite volume GCM

with NASA-GMAO: Todling, Errico, Gelaro, Winslow

- AD for fvGCM dynamical core (Lin and Rood, 1996; Lin, 1997)
- ~ 87'000 lines of Fortran 90 (without comments)
- Parallelisation: Message Passing Interface (MPI) + OpenMP
- Tangent Linear and adjoint generated by TAF
  only hand written code for adjoint MPI wrappers
  OpenMP handled by TAF
- Adjoint can use 2 level checkpointing
- uses features such as
  free source form, direved types, allocatable arrays
- good performance TLM and ADM crucial for applications
- To be used by GMAO for
  - Data assimilation,
  - Sensitivity studies,
  - Singular vector detection ...

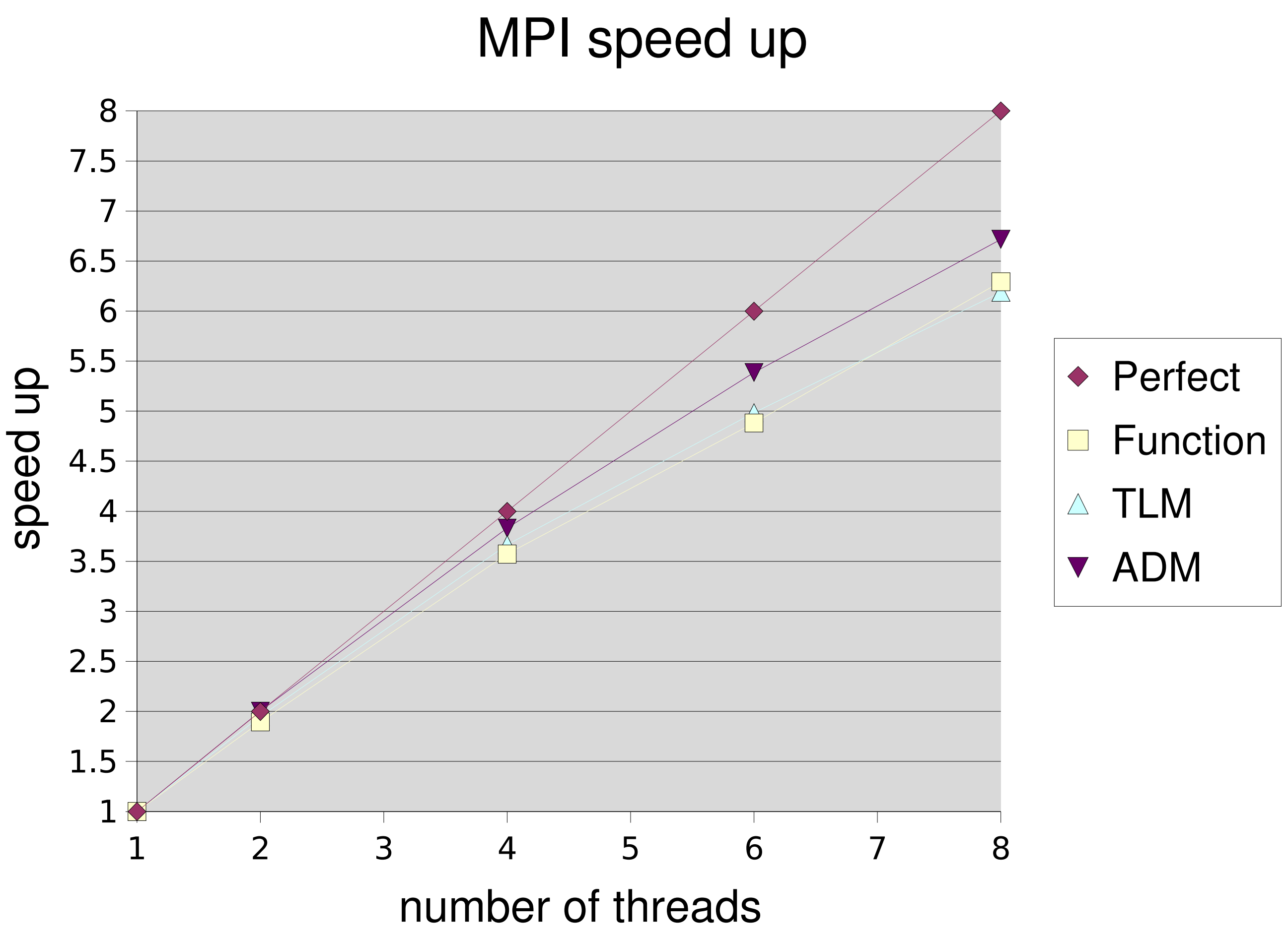
AD of fvGCM
Exploiting TAF flow directives

- TLM and ADM need to linearise around external trajectory
  - Function code overwrites state
  - data flow from initial to final state interrupted
  - straight forward use of AD results in erroneous derivatives
  - Exploit TAF's flexibility in generation of store/read scheme:
    trigger generation of desired behaviour
    by combination of TAF init and store directives
  - Generated code is, however, not derivative of function code
- Code uses FFT and its inverse
  - Reusing FFT in TLM and inverse FFT for ADM is more efficient than differentiating FFT (Giering et al, 2002)
  - Reuse triggered by TAF flow directives

AD of fvGCM
Handling MPI

- Model has wrapper routines (e.g. mp_send3d_ns) that call the respective MPI library routines (e.g. mpi_isend)
- Wrappers are encapsulated in one module
- Decision between MPI-1/2 happens in wrappers
- In forward mode, TAF handles (most) MPI calls.
  We need, however, TLM and ADM
  -> Construction of MPI in TLM and ADM at level of wrappers
    - Inserting of TAF flow directives for wrappers
    - TLM and ADM wrapper routines hand written
    - TLM and ADM wrappers reuse model wrappers (easy to maintain)
    - Handling of MPI-1 and MPI-2 at once
- Encapsulation helped a lot!

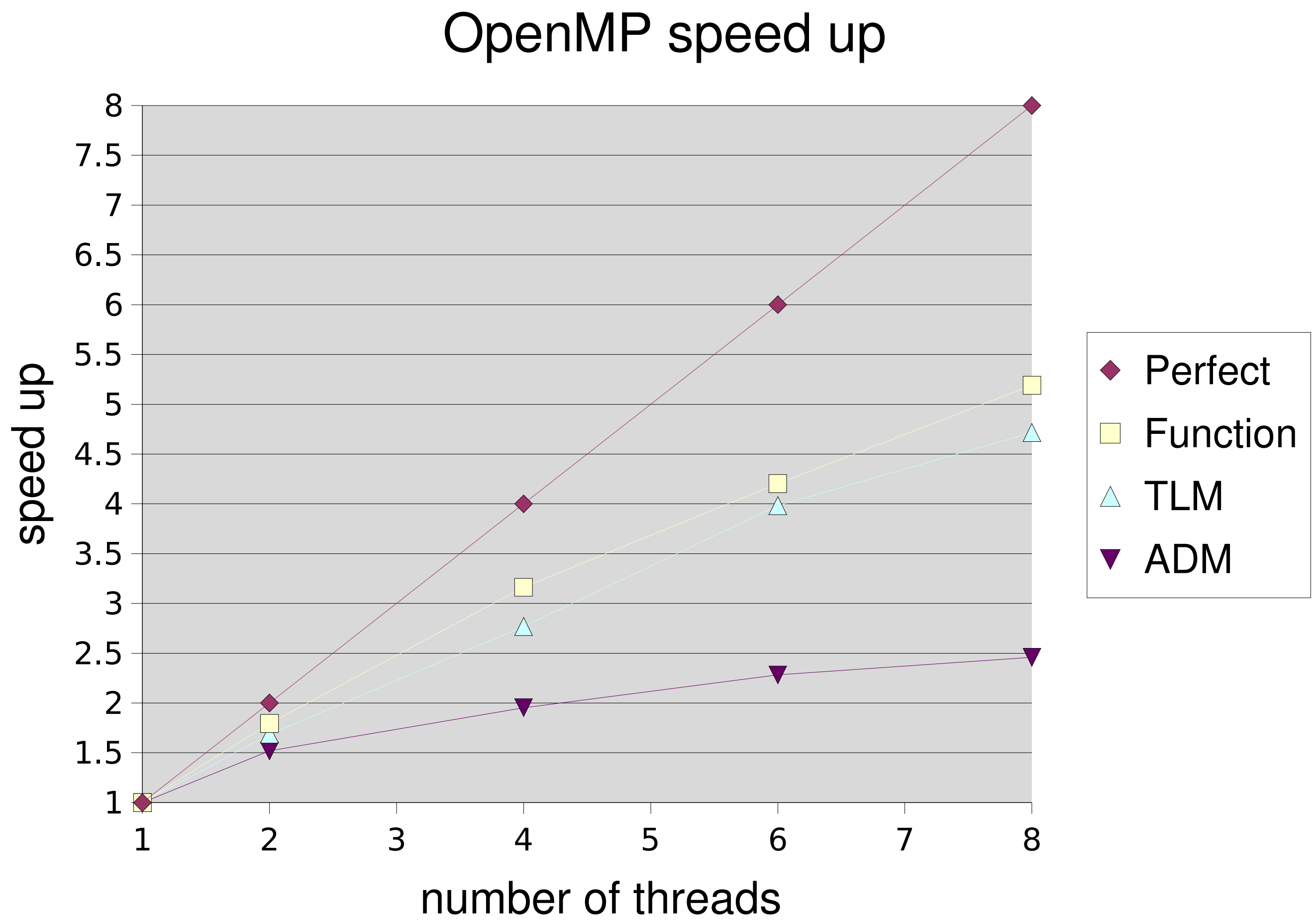
MPI


MPI speed up
8
7.5
7
6.5
6
5.5
5
4.5
4
3.5
3
2.5
2
1.5
1
1
2
3
4
5
6
7
8
speed up
number of threads
Perfect
Function
TLM
ADM

AD of fvGCM
Handling of OpenMP

- Model uses only a single directive:
  !$omp parallel do
- TAF analyses the loop-carried dependencies
- For ADM loop, according to the dependencies, TAF generates the proper !$omp directive for the adjoint loop and (if necessary) additional statements to preserve parallelism
- Can generate code for OpenMP-1 or OpenMP-2
- OpenMP-1 adjoint of fvGCM need many critical sections, because OpenMP-1 does not support array reductions.
- OpenMP-2 does and thus yields faster code.
- For TLM loop, TAF uses the similar directive

OpenMP-1


OpenMP speed up
speed up
8
7.5
7
6.5
6
5.5
5
4.5
4
3.5
3
2.5
2
1.5
1
1
2
3
4
5
6
7
8
number of threads
Perfect
Function
TLM
ADM

TAF
Transformation of Algorithms in Fortran

- **Source-to-source translator for Fortran-77/90/95**
- **forward and reverse mode**
- **scalar and vector mode**
- **full and pure mode**
- **efficient Hessian code by applying TAF twice (e.g. forward over reverse)**
- **command line program with many options**
- **TAF-Directives are Fortran comments**
- **extensive and complex code analyses (similar to optimising compilers)**
- **generated code is structured and well readable**

TAF
More features

- Generation of flexible store/read scheme for required values triggered by TAF init and store directives
- Generation of simple checkpointing scheme (Griewank, 1992) triggered by combination of TAF init and store directives
- Generation of efficient adjoint (Christianson, 1996, 1998) for converging iterations triggered by TAF loop directive
- TAF flow directives for black-box routines, or to include user provided derivative code (exploit linarity or self-adjointness, MPI wrappers, etc...)
- Automatic Sparsity Detection
- Basic support for MPI and OpenMP
- supports interrupting and restarting adjoint ('divided adjoint')

TAF
support of Fortran-95

- **supported:**
  - all intrinsic functions (SUM,CSHIFT,TRANSPOSE,NULL,etc.)
  - WHERE, SELECT
  - derived types
  - generic functions
  - recursive, pure, elemental functions
  - private variables, interfaces
- **with restrictions:**
  - pointers
  - allocation, deallocation
  - FORALL
- **not yet supported:**
  - operator overloading

some larger TAF Derivatives

Model (Who)	Lines	Lang	TLM	ADM	Ckp	HES
NASA/GMAO (w. Todling et al.)	87'000	F90	1.5	7.0	2 lev	-
MOM3 (Galanti & Tziperman)	50'000	F77	Yes	4.6	2 lev	-
MITGCM (ECCO Consortium)	100'000	F77	1.8	5.5	3 lev	11.0/1
BETHY (w. Knorr, Rayner, Scholze)	5'400	F90	1.5	3.6	2 lev	12.5/5
Nav.-Stokes-Solver (Hinze, Slawig)	450	F77	-	2.0	steady	-
NSC2KE (w. Slawig)	2'500	F77	2.4	3.4	steady	9.8/1
HB_AIRFOIL (Thomas & Hall)	8'000	F90	-	3.0		-
ARPS (Yang, Xue, Martin) in progress	40'000	F90	2.0	11.0	2 lev	-
NIRE-CTM	860	F77	1.0	1.5		-

- Lines: total number of Fortran lines without comments
- Numbers for TLM and ADM give CPU time for (function + gradient) relative to forward model
- HES format: CPU time for Hessian * n vectors rel. t. forw. model/ n
- 2 (3) level checkpointing costs 1 (2) additional model run(s)

Summary

- TLM and ADM of large Fortran 77/90 codes
- TAF handles almost full Fortran-95 standard
- retain parallelisation in derivative code (OpenMP and MPI)
- TAF can update derivative code in one-click procedure
- performance of tangent, adjoint and Hessian codes is good
- AD helps to reduce the delay from model development to data assimilation and related applications
- Concepts are being transferred from Fortran to C

  (see next talk)