Generating the tangent linear and adjoint models of the DAO finite volume GCM's dynamical core by means of TAF

R. Giering[1], T. Kaminski[1], R. Todling[2], and S.-J. Lin[2]

[1] **FastOpt, Hamburg ;** [2] **NASA/GSFC Data Assimilation Office, Greenbelt, Md**

Find copy of poster at: http://www.FastOpt.com

Intro

We are applying FastOpt's automatic differentiation tool Transformation of Algorithms in Fortran (TAF, *Giering and Kaminski* [1998]) to the DAO finite volume General Circulation Model's (fvGCM's, *Lin and Rood* [1997]) dynamical core. TAF automatically generates the tangent linear and adjoint models from the GCM's original Fortran 90 source code. The TLM and ADM will be used by the DAO in the retrospective analysis system [*Zhu et al.*, 2002], singular vector analyses, and sensitivity studies. TAF also generates a version of the TLM/ADM, which is restricted to propagation of tracer perturbations.

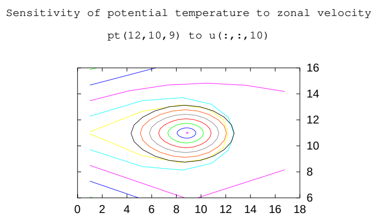

Figure: Example of an adjoint sensitivity calculation.

TAF

- Source-to-source translator for Fortran-77 to 95
- Commercial successor of TAMC
- Generates tangent linear (TLM) and adjoint models (ADM)
- Generates Hessian (2nd derivative) code by applying TAF twice (forward over reverse mode)
- Flexibility trough command line options and TAF-Directives
- Extensive and complex code analyses (like optimising compilers)
- Generated code is structured and well readable

Code generation

- Needed modifications of fvGCM code + TAF enhancements
- Generation process fully automated
- TAF Flow directives to use hand written MPI wrappers and to reuse FFT in TLM/ADM
- Flexible storing/reading scheme for required variables (trajectory) via TAF directives
- Checkpointing scheme for large integrations via TAF directives
- OpenMP and MPI parallelism preserved in TLM and ADM
- ADM will be optimised for production platforms.

Preserving MPI parallelism

fvGMC uses wrappers which call MPI routines, code example exchanges values of field q.

```
call mp_send4d_ns(im,  jm,  jfirst,  jlast,  1,  km,  nq,  ng,  ng,  q)
... (computations)  ...
call mp_recv4d_ns(im,  jm,  jfirst,  jlast,  1,  km,  nq,  ng,  ng,  q)
```

ADM-wrappers and some TLM-wrappers hand written. Used TAF flow directives to include TLM/ADM-wrappers. TLM can reuse fvGCM wrapper to send q and its perturbation q_tl.

```
call mp_send4d_ns(  im,jm,jfirst,jlast,1,km,nq,ng,ng,q_tl  )
call mp_send4d_ns(  im,jm,jfirst,jlast,1,km,nq,ng,ng,q  )
... (model and tangent linear computations)  ...
call mp_recv4d_ns(  im,jm,jfirst,jlast,1,km,nq,ng,ng,q_tl  )
call mp_recv4d_ns(  im,jm,jfirst,jlast,1,km,nq,ng,ng,q  )
```

ADM needs hand written wrapper to send/receive q_ad.

```
call mp_recv4d_ns_ad(  im,jm,jfirst,jlast,1,km,nq,ng,ng,q_ad  )
... (adjoint computations)
call mp_send4d_ns_ad(  im,jm,jfirst,jlast,1,km,nq,ng,ng,q_ad  )
```

Preserving OpenMP parallelism

OpenMP directives are instructions to the Fortran compiler. fvGCM example triggers parallelisation over the vertical layers:

```
!$omp parallel do private(i,  j, k)
      do k=1,km
        do j=js1g1,jn1g1                  ! dpt needed NS
          do i=1,im                       ! wz, pkc ghosted NS
            dpt(i,j,k)=(wz(i,j,k+1)+wz(i,j,k))*(pkc(i,j,k+1)-pkc(i,j,k))
          enddo
        enddo
      enddo
```

TAF analyses loop structure and writes OpenMP directives into TLM and ADM code. This is the OpenMP 1 version of example (OpenMP 2 looks nicer):

```
pkc_adh(:,:,:)  = pkc_ad(:,:,:)
wz_adh(:,:,:)  = wz_ad(:,:,:)
!$omp parallel do shared(dpt_ad,im,jn1g1,js1g1,km,pkc,pkc_adh,wz,wz_adh) private(i,j,k,pkc_ad,wz_ad)
do k = 1, km
  pkc_ad(:,:,:)  = 0.d0
  wz_ad(:,:,:)  = 0.d0
  do j = js1g1, jn1g1
    do i = 1, im
      pkc_ad(i,j,k+1)  = pkc_ad(i,j,k+1)+dpt_ad(i,j,k)*(wz(i,j,k+1)+wz(i,j,k))
      pkc_ad(i,j,k)  = pkc_ad(i,j,k)-dpt_ad(i,j,k)*(wz(i,j,k+1)+wz(i,j,k))
      wz_ad(i,j,k+1)  = wz_ad(i,j,k+1)+dpt_ad(i,j,k)*(pkc(i,j,k+1)-pkc(i,j,k))
      wz_ad(i,j,k)  = wz_ad(i,j,k)+dpt_ad(i,j,k)*(pkc(i,j,k+1)-pkc(i,j,k))
      dpt_ad(i,j,k)  = 0.d0
    end do
  end do
  !$omp critical  (adtaf)
  pkc_adh(:,:,:)  = pkc_adh(:,:,:)+pkc_ad(:,:,:)
  wz_adh(:,:,:)  = wz_adh(:,:,:)+wz_ad(:,:,:)
  !$omp end critical (adtaf)
end do
pkc_ad(:,:,:)  = pkc_adh(:,:,:)
wz_ad(:,:,:)  = wz_adh(:,:,:)
```

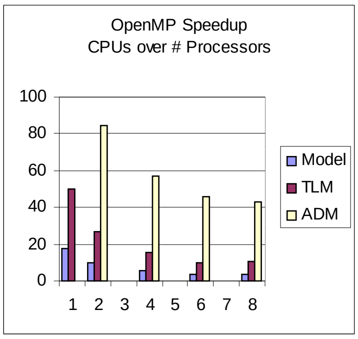

Figure: OpenMP speedup of GCM/TLM/ADM.

References

Giering, R., and T. Kaminski, Recipes for Adjoint Code Construction, *ACM Trans. Math. Software*, *24*, 437–474, 1998.

Lin, S.-J., and R. B. Rood, An explicit flux-form semi-lagrangian shallow-water model on the sphere, *Quart. J. Roy. Meteor. Soc.*, *123*, 2477–2498, 1997.

Zhu, Y., R. Todling, J. Guoy, S. E. Cohn, I. M. Navon, and Y. Yang, The GEOS Retrospective Data Assimilation System: The 6 hour lag case, *to appear in Monthly Weather Review*, 2002.