Reduction of Temporal Discretization Error in an Atmospheric General Circulation Model

October 5, 2012

Daisuke Hotta
hotta (at) umd.edu

Advisor: Prof. Eugenia Kalnay
Department of Atmospheric and Oceanic Science
ekalnay (at) atmos.umd.edu

Abstract

An atmospheric general circulation model (AGCM) is a computer program which numerically integrates the system of partial differential equations which describes the physical laws governing the flow of the Earth's atmosphere. Integration of AGCM is a central part of weather predictions and climate projections. Currently, due to restrictions on computational costs, most AGCMs adopt relatively low-order temporal discretizations, under the premise that errors arising from temporal discretizations are negligible compared to those associated with spatial discretizations or physical parametrizations. However, recent study shows that temporal discretizations is likely to be a significant source of model errors. The goal of this project is to reduce the temporal discretization error. Two different approaches will be examined: the first approach is to test a new time integration scheme, called Lorenz N-cycle scheme, which requires the same computational costs as the conventional scheme but yet can yield forecasts of higher-order accuracy. The second approach is to empirically estimate and correct model errors using the methodology presented by Danforth et al. (2007).

1 Introduction

Current weather forecasts and climate projections are based on numerical integration of the fluid dynamical partial differential equations which govern the evolution of the global atmosphere. The models which discretize those governing equations of the global atmospheric motion are called **Atmospheric General Circulation Model**s, or **AGCM**s. Currently, most AGCMs, even the state-of-the-art, most comprehensive models, adopt rather simple, low-order temporal discretization schemes, including the second order leapfrog scheme (which falls into first order when used in conjunction with temporal filters such as Robert-Asselin filter (Asselin, 1972) for stabilization), or the first order Matsuno scheme (Matsuno, 1966). The rationale behind the use of such low-order schemes in AGCMs is that, the model errors are dominated by the components arising from spatial discretization or from physical parametrizations and thus, in order to strike the best balance between computational efficiency (both in terms of speed and memory consumption) and accuracy, temporal discretizations should be made economical.

However, given the trend of increased spatial resolutions fueled by increase of computing powers, and the accelerating betterment of physical parametrizations, it might be natural to cast doubt on the validity of the assumption that errors due to temporal discretizations are less significant, and to consider refinements to temporal integration schemes. In fact, recent research shows that outputs of AGCMs show sensitive dependence on time stepping. For example, Teixeira et al. (2007) has shown that NOGAPS, the US Navy Operational Global Atmospheric Prediction System, exhibits considerably different model climates when integrated with different time steps, indicating that temporal discretization errors can account for a substantial component of the total model errors. In this project, I will consider two possible remedies to the issue presented above. The approaches of the two remedies are outlined in the next section.

2 Approaches

The first approach is to use a temporal discretization scheme which is computationally as economical as the conventional schemes but yet can be substantially more accurate. The efficient and accurate scheme, called the Lorenz N-cycle scheme, will be implemented to an existing AGCM, called SPEEDY model, which adopts the leapfrog scheme with Robert-Asselin filter as its temporal discretization. As a reference, the tried-and-truth Runge-Kutta scheme of 4th-order will be implemented as well. The performance of the Lorenz N-cycle in terms of computational efficiency and accuracy will be compared to the original scheme (leapfrog) and the 4th-order Runge-Kutta scheme. The implementation, validation and verification of this approach will comprise the Phase I of this project.

The second approach is to estimate the model error and then reduce it by subtracting the estimated error during the course of model integration. For this purpose, I will reproduce the empirical error correction methods introduced by Danforth et al. (2007) who used model error bias statistics and the singular value decomposition (SVD) technique on the covariance matrix between model state and model error to estimate, respectively, the state-independent and the state-dependent component of the model error in the SPEEDY model. The methods will be applied both to the original SPEEDY model and to the newly implemented Lorenz N-cycle. The implementation, validation and verification of a subset of this approach will comprise the Phase II of this project.

3 Phase I

3.1 Lorenz N-cycle

In 1971 Edward Norton Lorenz devised an incredibly smart time-integration scheme for a system of first-order ordinary differential equations (ODEs) which achieves high-order accuracy and minimal memory consumption at the same time (Lorenz, 1971). Consider a problem of numerically integrating the system of ODEs:

$$\frac{du}{dt} = F(u) \tag{1}$$

where $u = (u_1(t), \ldots, u_M(t))$ is an M-dimensional vector function of t and $F(u) = (F_1(u_1, \ldots, u_M), \ldots, F_M(u_1, \ldots, u_M))$ is a function from $\mathbb{R}^M \to \mathbb{R}^M$. Lorenz (1971) derived two "isomeric" versions for the above problem whose algorithms are shown schematically as pseudo-codes below:

N-cycle A

```
    $w^0 = 1$,
    $w^k = \frac{N}{N-k}$ $(k = 1, \dots, N-1)$,
    do $k = 0, \dots$
        $w \leftarrow w^{\mathrm{mod}(k,N)}$
        $G \leftarrow wF(u) + (1-w)G$
        $u \leftarrow u + G\Delta t$
    end do
```

N-cycle B

```
    $w^0 = 1$,
    $w^k = \frac{N}{k}$ $(k = 1, \dots, N-1)$,
    do $k = 0, \dots$
        $w \leftarrow w^{\mathrm{mod}(k,N)}$
        $G \leftarrow wF(u) + (1-w)G$
        $u \leftarrow u + G\Delta t$
    end do
```

(Here I used the elegant notation introduced in Purser and Leslie (1997) rather than that in the original paper by Lorenz.)

Mathematical idea behind the above algorithms is simple: to "reuse" the previously computed tendencies $(F(u^{k-j}), j = 1, 2, \dots)$ by forming a weighted average of them to produce a tendency which yields the highest order of accuracy after the completion of N-th step, under the constraint that each intermediate step retains at least first order accuracy. In this sense, the Lorenz N-cycle can be regarded as a special type of Runge-Kutta family. In fact, it can be shown that for linear systems, the Lorenz 2-cycles, both version A and B, are equivalent to the Heun scheme with time step $2\Delta t$ and the Lorenz 4-cycles, both version A and B, are equivalent to the classical 4-step 4th-order Runge-Kutta scheme (described in the next subsection) with time step $4\Delta t$.

The two versions differ only in the choice of the "weight" coefficients w^k. Lorenz (1971) showed that, for a case where F is linear with respect to u, for any positive integer N, the both versions yield numerical solution of N-th order every N time steps, although in the intermediate steps, the accuracy is only of first order. If F is nonlinear, the accuracy of N-cycle reduces to 2nd order for $N \geq 3$. However, for $N = 3$ and for $N = 4$, the N-th order term in the truncation error of the versions A and B can be shown to be of same magnitude with opposite signs. Thus, for $N = 3$ and for $N = 4$, N-th order accuracy can be attained by running both A and B cycles simultaneously and averaging the predictions, at the expense of doubling the computational cost both in speed and memory consumption.

In order to avoid doubling of computational costs, Lorenz (1971) proposed to use the versions A and B in a suitably designed alternating sequence, based on the intuition that the errors the both versions should tend to cancel each other. In fact, he showed that, for $N = 3$, true 3rd order accuracy can be retained even for a nonlinear case by alternating versions A and B. Likewise, full 4th order accuracy can be achieved for $N = 4$ by forming a $4N(=16)$-cycle of A,B,B,A. In this project, I will implement this $4N(=16)$-cycle version and call it by the Lorenz N-cycle for convenience.

Advantages of Lorenz N-cycle

The principal advantage of Lorenz N-cycle beside the high-order accuracy is its computational efficiency, both in terms of speed and memory consumption. As is clear from the pseudo-code listed above, the scheme requires only one evaluation of $F(u)$ per time step which, in most cases, is the most computationally demanding part of the algorithm. Also, the scheme consumes only $2M$ words of memory. Thus, the Lorenz N-cycle has the same computational cost as the widely-used leapfrog scheme. Compared to the 4th-order Runge-Kutta scheme which is very accurate but is impractically too expensive for the purpose of AGCMs, the Lorenz N-cycle consumes less than half memory and can run 4 times faster.

Another great advantage of the Lorenz N-cycle is the absence of computational modes: the Lorenz N-cycle, being a two time-level method (i.e., the state at time t alone completely determines the state at time $t + \Delta t$) rather than a three time-level method like leapfrog scheme, does not suffer from the presence of computational mode. This feature proves to be particularly useful for nonlinear systems for which computational modes tend to grow causing divergence of numerical solution from the physical solution.

Being two-time level scheme also facilitates the initialization process for which, in the case of three or more time-level schemes, a special treatment of the very first step(s) are required.

Despite these practical merits, the Lorenz N-cycle scheme has remained "forgotten". Although there are some oceanic models which uses this method as its temporal discretization, and Purser and Leslie (1997) developed and tested a scheme which they devised inspired by the Lorenz N-cycle scheme, no direct application has been made to AGCMs.

3.2 4th-order Runge-Kutta scheme

As a reference, I will also implement the tried-and-tested classical 4th-order Runge-Kutta scheme. In a pseudo-code, its algorithm is:

```
**do** $k = 0, \dots,$
    $h^1 \leftarrow F(u)$
    $h^2 \leftarrow F\left(u + \frac{\Delta t}{2} h^1\right)$
    $h^3 \leftarrow F\left(u + \frac{\Delta t}{2} h^2\right)$
    $h^4 \leftarrow F\left(u + \Delta t h^3\right)$
    $u \leftarrow \frac{\Delta t}{6}\left(h^1 + 2h^2 + 2h^3 + h^4\right)$
**end do**
```

As we can clearly see, this algorithm requires 4 evaluations of F per time step and $5M$ words of memory.

3.3 The SPEEDY model

In this project, the Lorenz N-cycle scheme and the Runge-Kutta 4th-order scheme will be implemented to an existing AGCM known as the SPEEDY (simplified parametrizations, primitive equation dynamics) model (Molteni, 2003). It is a primitive equation model with 8 vertical layers in σ-coordinate whose horizontal discretization is spectral representation with respect to spherical harmonics triangularly truncated at total wave number of 30 (T30). As its temporal discretization, leapfrog scheme is used for the dynamics but for the physical parametrizations, 1st order Forward Euler scheme with time step of $2\Delta t$ is used. The Robert-Asselin filter is also applied. Namely, in a pseudo-code, the algorithm is:

```
**do** $k = 1, \dots$
    $u^{+1} \leftarrow u^{-1} + 2\Delta t(F_{\mathbf{Dyn}}(u^0) + F_{\mathbf{Phys}}(u^{-1}))$
    $u^0 \leftarrow u^0 + \alpha(u^{+1} - 2u^0 + u^{-1})$
    $t \leftarrow t + \Delta t$
    $u^{-1} \leftarrow u^0, \quad u^0 \leftarrow u^{+1}$
**end do**
```

Semi-implicit treatment of the fastest external gravity waves is also applied which enables stable integration with a large value of Δt by circumventing the CFL condition associated with external gravity waves. The default time step for the SPEEDY model is $\Delta t = 40$min.

Details of the prognostic variables (u) and the equations solved by the model, including complete specification of the forcing $F^{\mathbf{Dyn}}(u)$ and an outline of the forcing $F^{\mathbf{Phys}}(u)$ are described in the Appendix 1.

Simplified physical process and coarse resolution enable the SPEEDY model to be integrated very fast. Despite such simplification, this model is able to produce realistic simulations of a wide range of the atmospheric phenomena, including mid-latitude synoptic features and climatology.

The SPEEDY model is written in Fortran 77 and can be run on virtually any machine which supports Fortran 77 compiler. A Linux server hosted by the AOSC department will be used in this project.

3.4 Validation

Validation of the implementation will be conducted by carrying out the bench mark for AGCMs called "Jablonowski-Williamson dynamical core test"(Jablonowski and Williamson, 2006). The test consists of two test cases. The first one is the "steady-state test case" in which the model is run from an analytic steady-state initial condition. The validity of the model is judged based on to what extent the model can maintain the steady state.

In the second test case, called the "baroclinic wave test case", the steady-state initial condition is perturbed so that the model, if run from this initial condition, will yield baroclinic wave. Baroclinic waves are the unstable waves which drive extratropic synoptic weather disturbances. Hence, successful simulation of baroclinic waves is of crucial importance for AGCMs. Unfortunately, however, no analytic solution is known for the baroclinic wave test case. In lieu of this, Jablonowski and Williamson (2006) provides a reference solution which is generated from four distinct high-resolution (approximately corresponds to 50km-mesh) AGCMs. The uncertainty estimate of the reference solution is also provided which are evaluated as the differences among those high-resolution models. The data of reference solution along with its uncertainty estimate is publicly available from the University of Michigan website:

```
http://esse.engin.umich.edu/groups/admg/ASP_Colloquium.php
http://www-personal.umich.edu/~cjablono/dycore_test_suite.html
```

In order to conduct the above test cases, I will first switch-off physical parametrizations from the SPEEDY model. This can be done by simply commenting-out a call to the driver subroutine of the physics package. Second, the orography (mountains) will be made flat. Since the orography is implemented as an external input to the SPEEDY model, removal of orography can be done simply by making a new orography file containing zeros for all grids on the Earth.

The results of baroclinic test case obtained for the the newly-implemented Lorenz N-cycle and 4th-order Runge-Kutta scheme, along with the original leapfrog scheme, will be compared to the provided reference solution. If the newly-integrated schemes are closer or equally close to the reference solution, I conclude that the implementation has been made successfully.

The set-ups of the experiments are summarized in the following table:

ExpID	Scheme	Initial Condition	Reference Data
STDY_LF	Leapfrog	Analytic steady-state	Initial condition itself.
STDY_NCYC	N-cycle		
STDY_RK4	Runge-Kutta 4		
BRCL_LF	Leapfrog	Steady-state superposed with a disturbance	Reference Solution.
BRCL_NCYC	N-cycle		
BRCL_RK4	Runge-Kutta 4		

3.5 Verification

The validation described in the previous section only evaluates the validity of the newly-implemented schemes in a specific configuration. Notably, the validation procedure does not evaluate the performance of the schemes in conjunction with physical parametrizations.

In order to evaluate the full performance of the newly-implemented schemes, I will produce and plot model climatology and compare them with the official plots by the developers published at the following web site:

`http://users.ictp.it/~kucharsk/speedy8_clim.html`

4 Phase II

In Phase II of this project, the training stage of Danforth et al. (2007) will be reproduced, both for the original SPEEDY model with leapfrog scheme and for the SPEEDY model with the newly implemented schemes. The second stage of Danforth et al. (2007), the estimation-correction stage, will be addressed in Phase III of this project provided that time permits. For the outline of Phase III, see the Appendix 2.

4.1 Algorithm

The first step of the training stage of Danforth et al. (2007) is to generate many samples of model errors. First, a 6-hour forecast is performed by running the SPEEDY model with the original leapfrog scheme, denoted $M_{6h}^{LF}(x)$ here after, from the initial condition taken from the National Centers for Environmental Prediction (NCEP)/National Center for Atmospheric Research (NCAR) reanalysis (Kalnay and Coauthors, 1996), which will be denoted $x^{true}(t)$ here after. The discrepancy between the forecast $M_{6h}^{LF}(x^{true}(t))$ and the corresponding atmospheric state from NCEP/NCAR reanalysis $x^{true}(t + 6\text{hours})$, namely

$$\delta x(t) = M_{6h}^{LF}(x^{true}(t)) - x^{true}(t + 6\text{hours}),$$

is regarded as the model error. The 6-hour forecasts are repeated many times using different initial conditions to collect many samples of model errors $\delta x(t)$.

The next step in the training stage is to extract state-independent component, or the bias, of the model error. The bias is estimated as the mean of the sampled model errors $\langle \delta x \rangle$, where the angle brackets denotes averaging over all samples.

The third step is to build the covariance matrix between the model state and the model error:

$$C = \left\langle \left(x^{true}(t) - \langle x^{true} \rangle\right) \left(\delta x(t) - \langle \delta x \rangle\right)^T \right\rangle .$$

In prior to building the covariance matrix, the mean will be removed, both for the model state and the model error.

Finally, to extract the dominant modes of the co-variation of the two quantities, singular value decomposition (SVD) will be performed on the matrix C:

$$C = U \Sigma V^T$$

where U and V are square orthogonal matrices and Σ is a diagonal matrix. The right singular vector v_i can be interpreted as the shape of model error which is most likely to be assumed if the anomaly of the model state $(x^{true}(t) - \langle x^{true} \rangle)$ is in the direction of the corresponding left singular vector u_i. Thus, pairs of left and right singular vectors provide valuable information about the characteristics of the model error. In the estimation-correction stage of Danforth et al. (2007), which, time permitted, will be addressed in Phase III of this project, the singular vectors and the singular values will be used to build a regression model with which we can estimate the most likely state-dependent component of model error given the current state of the model.

In this project, the above procedures will be repeated for the SPEEDY model with the original leapfrog scheme (M^{LF}), the Lorenz N-cycle scheme (M^{Ncyc}) and that with the 4th-order Runge-Kutta scheme (M^{RK4}).

The algorithm can be summarized as the following:

```
for models $M$ in $M^{LF}, M^{Ncyc}, M^{RK4}$, do
  do $i = 1, 2, \dots$
    - perform forecast $x^{fcst}(t_i + 6\text{hours}) = M_{6h}(x^{true}(t_i))$
    - compute the error $\delta x(t_i) = x^{fcst}(t_i + 6\text{hours}) - x^{true}(t_i + 6\text{hours})$
  end do
  - compute the bias $\langle \delta x \rangle = \frac{1}{\#i} \sum_i \delta x(t_i)$
  - compute the climatology $\langle x^{true} \rangle = \frac{1}{\#i} \sum_i x^{true}(t_i)$
  - compute the covariance matrix
      $C = \frac{1}{\#i-1} \sum_i \left(x^{true}(t_i) - \langle x^{true} \rangle\right) \left(\delta x(t_i) - \langle \delta x \rangle\right)^T$
  perform SVD for $C$:   $C = U \Sigma V^T$
end do
```

The NCEP/NCAR reanalysis data which will be used as truth are publicly available from NCEP or Earth System Research Laboratory (ESRL) of National Oceanic and Atmospheric Administration (NOAA) at:

`http://www.esrl.noaa.gov/psd/data/gridded/data.ncep.reanalysis.html`
`http://www.nomad3.ncep.noaa.gov/ncep_data/`

4.2 Implementation and Platform

Programs to be implemented in Phase II are the following:

1. a program to compute the bias
2. a program to compute the covariance matrix C
3. a program to perform SVD on C

These programs will be implemented in Fortran 90 on a Linux server hosted on the network of the AOSC department.

4.3 Validation

For the SVD code, validation will be conducted by first preparing a small-dimensional dummy data for the covariance and then performing the SVD on this dummy data both by running the implemented code and by running a Matlab's package routine. The implementation is judged successful if the two results agrees.

The entire implementation will be judged successful if the bias and singular vectors obtained for M^{LF} agrees with those published in Danforth et al. (2007).

4.4 Verification

In the verification of Phase I, the accuracy of Lorenz N-cycle will be tested only for the dynamical core of the AGCM. In the verification process of Phase II, its accuracy as a comprehensive AGCM including physical parametrization will be tested by comparing the amplitude of model error extracted as the bias and covariance with those for the original model. If the amplitudes are smaller for the Lorenz N-cycle than for the original leapfrog scheme, it means that the model with Lorenz N-cycle is more accurate.

5 Deliverables

Deliverables of Phase I are:

1. subroutines for Lorenz N-cycle and 4th-order Runge-Kutta of the SPEEDY model
2. results of Jablonowski-Williamson dynamical core test cases for the SPEEDY model, both with the original leapfrog scheme and the newly implemented schemes.
3. plots of model climatology for the SPEEDY model, both with the original leapfrog scheme and the newly implemented schemes.

Deliverables of Phase II are:

1. an archive of the model errors
2. model error bias
3. pairs of singular vectors for the model state and the model error along with the corresponding singular values

4. code for performing SVD.

Deliberables also includes

1. two presentations, mid-year and final
2. two reports, mid-year and final
3. and this project proposal.

6 Schedule and Milestones

Schedule for the project is as follows:

- Phase I
  - Implement Lorenz N-cycle and 4th-order Runge-Kutta scheme to the SPEEDY model: now through end of November.
  - Write the mid-year report, prepare the oral presentation: December
  - Switch-off physical parametrizations and prepare flat orography: January
  - Perform the dynamical core tests: early February
- Phase II
  - Generate initial values from the NCEP/NCAR reanalysis: end of February
  - Build the bias and covariance matrix: March
  - Code and test a program for SVD: April
  - Compare the model errors for M^{LF} with those published in Danforth et al. (2007): early May
  - Compare the model errors for M^{LF} and M^{Ncyc}, M^{RK4}: mid-May
  - Write the final report and prepare for the oral presentation.

The corresponding milestones are:

- Phase I
  - The SPEEDY model with Lorenz N-cycle and 4th-order Runge-Kutta scheme both runs: end of November

  - Complete validation of Lorenz N-cycle and 4th-order Runge-Kutta scheme: February

- Phase II
  - Complete computation of error bias and covariance: March
  - Complete validation of the SVD program: April
  - Complete validation of the entire procedure: early May
  - Complete verification: mid-May

References

Asselin, R., 1972: Frequency filter for time integrations. *Mon. Wea. Rev*, **100**, 487–490.

Danforth, M., C, E. Kalnay, and T. Miyoshi, 2007: Estimating and correcting global weather model error. *Mon. Wea. Rev*, **135**, 281–299.

Jablonowski, C. and D. L. Williamson, 2006: A baroclinic instability test case for atmospheric model dynamical cores. *Q. J. R. Meterol. Soc*, **132**, 2943–2975.

Kalnay, E. and Coauthors, 1996: The ncep/ncar 40-year reanalysis project. **77**, 437–471.

Lorenz, N., E., 1971: An n-cycle time-differencing scheme for step-wise numerical integration. *Mon. Wea. Rev*, **119**, 1612–1623.

Matsuno, T., 1966: Numerical integrations of the primitive equations by a similated backward difference method. *Bull. Amer. Meteor. Soc*, **44**, 76–84.

Molteni, F., 2003: Atmospheric simulations using a gcm with simplified physical parametrizations. i: Model climatology and variability in multi-decadal experiment. *Climate Dyn.*, **20**, 175–191.

Purser, J., R. and L. M. Leslie, 1997: High-order generalized lorenz n-cycle schemes for semi-lagrangian models employing second derivatives in time. *Mon. Wea. Rev*, **125**, 1261–1276.

Teixeira, J., C. A. Reynolds, and K. Judd, 2007: Time step sensitivity of nonlinear atmospheric models: Numerical convergence, truncation error growth, and ensemble design. *J. Atmos. Sci*, **64**, 175–189.

Appendix 1: Detailed Description of the SPEEDY model

A1.1 The equations

The SPEEDY model is based on the nonlinear primitive equations for moist atmosphere on a vertical σ-coordinate with spherical geometry. Its prognostic variables are horizontal vorticity ζ, horizontal divergence D, temperature T, specific humidity q and the natural logarithm of surface pressure $\pi \equiv \ln p_s$. These equations are vertically discretized by a finite differencing and then horizontally discretized by Galerkin spectral method with respect to spherical harmonics. Namely, horizontal structure of any variable on a given vertical level is represented expansion coefficients of spherical harmonics, and horizontal differentiations are performed with respect to such coefficients. Thus, the actual prognostic variables in the computer code (expressed as u in the pseudo-codes in Section 3) are therefore the expansion coefficients of the above 5 prognostic variables on specified vertical levels.

The set of prognostic equations can be written explicitly as follows:

$$\frac{\partial \zeta}{\partial t} = \frac{1}{a(1-\mu^2)}\frac{\partial \mathcal{F}_V}{\partial \lambda} - \frac{1}{a}\frac{\partial \mathcal{F}_U}{\partial \mu} - K_\nu \left(\nabla_\sigma^4 - \frac{2^2}{a^2}\right)\zeta + \left.\frac{d\zeta}{dt}\right|_{\mathbf{Phys}} \quad (A.1)$$

$$\begin{aligned}\frac{\partial D}{\partial t} = \frac{1}{a(1-\mu^2)}\frac{\partial \mathcal{F}_U}{\partial \lambda} + \frac{1}{a}\frac{\partial \mathcal{F}_V}{\partial \mu} \\ - \nabla_\sigma^2(\Phi + R\bar{T}\pi + KE) - K_\nu \left(\nabla_\sigma^4 - \frac{2^2}{a^2}\right) D\end{aligned} \quad (A.2)$$

$$\begin{aligned}\frac{\partial T}{\partial t} = & -\frac{1}{a(1-\mu^2)}\frac{\partial UT'}{\partial \lambda} - \frac{1}{a}\frac{\partial VT'}{\partial \mu} + T'D \\ & -\dot{\sigma}\frac{\partial T}{\partial \sigma} + \kappa T\left(\frac{\partial \pi}{\partial t} + \boldsymbol{v}_H \cdot \nabla_\sigma \pi + \frac{\dot{\sigma}}{\sigma}\right) \\ & -K_h\left(\nabla_\sigma^4 - \frac{2^2}{a^2}\right)T + \left.\frac{dT}{dt}\right|_{\mathbf{Phys}}\end{aligned} \quad (A.3)$$

$$\frac{\partial q}{\partial t} = -\boldsymbol{v}_H \cdot \nabla_\sigma q - \dot{\sigma}\frac{\partial q}{\partial \sigma} + \left.\frac{dq}{dt}\right|_{\mathbf{Phys}} \quad (A.4)$$

$$\frac{\partial \pi}{\partial t} = -\boldsymbol{v}_H \cdot \nabla_\sigma \pi - \frac{\partial \dot{\sigma}}{\partial \sigma} - D \quad (A.5)$$

where

$$\theta \equiv T\,(p/p_0)^{-\kappa} \tag{A.6}$$

$$\kappa \equiv R/C_p \tag{A.7}$$

$$\Phi \equiv gz \tag{A.8}$$

$$= \Phi|_{\sigma=1} - \int_1^\sigma \frac{RT}{\sigma} d\sigma \tag{A.9}$$

$$\pi \equiv \ln p_S \tag{A.10}$$

$$\dot\sigma \equiv \frac{d\sigma}{dt} \tag{A.11}$$

$$\mu \equiv \sin\varphi \tag{A.12}$$

$$U \equiv u\cos\varphi \tag{A.13}$$

$$V \equiv v\cos\varphi \tag{A.14}$$

$$\zeta \equiv \frac{1}{a(1-\mu^2)}\frac{\partial V}{\partial\lambda} - \frac{1}{a}\frac{\partial U}{\partial\mu} \tag{A.15}$$

$$D \equiv \frac{1}{a(1-\mu^2)}\frac{\partial U}{\partial\lambda} + \frac{1}{a}\frac{V}{\mu} \tag{A.16}$$

$$\mathcal{F}_U \equiv (\zeta+f)V - \dot\sigma\frac{\partial U}{\partial\sigma} - \frac{RT'}{a}\frac{\partial\pi}{\partial\lambda} \tag{A.17}$$

$$\mathcal{F}_V \equiv -(\zeta+f)U - \dot\sigma\frac{\partial V}{\partial\sigma} - \frac{RT'}{a}(1-\mu^2)\frac{\partial\pi}{\partial\mu} \tag{A.18}$$

$$KE \equiv \frac{U^2+V^2}{2(1-\mu^2)} \tag{A.19}$$

$$\boldsymbol{v}_H\cdot\nabla \equiv \frac{u}{a\cos\varphi}\left(\frac{\partial}{\partial\lambda}\right)_\sigma + \frac{v}{a}\left(\frac{\partial}{\partial\varphi}\right)_\sigma$$

$$= \frac{U}{a(1-\mu^2)}\left(\frac{\partial}{\partial\lambda}\right)_\sigma + \frac{V}{a}\left(\frac{\partial}{\partial\mu}\right)_\sigma \tag{A.20}$$

$$\nabla_\sigma^2 \equiv \frac{1}{a^2(1-\mu^2)}\frac{\partial^2}{\partial\lambda^2} + \frac{1}{a^2}\frac{\partial}{\partial\mu}\left[(1-\mu^2)\frac{\partial}{\partial\mu}\right]. \tag{A.21}$$

Here, θ in Eq.(A.6) denotes potential temperature, Φ in (A.8) denotes geopotential height, φ and λ denote, respectively, latitude and longitude, a denotes the radius of the Earth, and KE in (A.19) denotes the kinetic energy per unit mass.

In some terms, temperature is divided into the horizontal global mean and the deviation therefrom:

$$T \equiv \bar{T}(\sigma) + T' \tag{A.22}$$

Vertical wind speed in the σ coordinate $\dot{\sigma}$ can be diagnosed as

$$\dot{\sigma} = -\sigma \frac{\partial \pi}{\partial t} - \int_0^\sigma D d\sigma - \int_0^\sigma \boldsymbol{v}_H \cdot \nabla_\sigma \pi d\sigma, \tag{A.23}$$

A Crank-Nicolson-type semi-implicit scheme is applied to filter-out external gravity waves with fast phase speed, whereby enabling an efficient long time-stepping in the integration.

A1.2 ”Dynamics” and “Physics”

In the literature of meteorology, the tendencies of prognostic equations are conventionally divided in to “dynamics” part and “physics” part. The physics part, denoted by $F_{\mathbf{Phys}}(u)$ in the pseudo-code in Section 3.3, is comprised of the terms of the form $\left.\frac{d(\cdot)}{dt}\right|_{\mathbf{Phys}}$ in Eq. (A.1)-(A.5). The dynamics part, denoted by $F_{\mathbf{Dyn}}(u)$ in the pseudo-code in Section 3.3, corresponds to all the terms in the right hand side of Eq. (A.1)-(A.5) except the physics part defined.

In the SPEEDY model, the physics tendencies $F_{\mathbf{Phys}}(u)$ includes contributions from:

- Convection (cumulonimbus)
- Large-scale condensation and Clouds
- Shortwave and Longwave radiation
- Surface fluxes of momentum and energy
- and Vertical diffusion (planetary boundary layer (PBL)).

The details for these parametrizations can be found in the model description by the developers which is available at

`http://users.ictp.it/~kucharsk/speedy-net.html`

A1.3 Boundary conditions

Boundary conditions used in the SPEEDY model are:

- Orography, expressed as the geopotential height at the surface ($\Phi|_{\sigma=1}$)
- land-sea mask

- sea surface temperature (SST)
- sea ice fraction
- soil temperature
- snow depth
- bare-surface albedo
- vegetation coverage

The first two and the last two remain constant during the integration. The remaining fields are given to the model as prescribed climatologies and vary with seasonal march. These boundary conditions are input to the model from external files.

A1.4 Control of Boundary Conditions in the Validation

For the validation of Phase I described in Section 3.4, physics tendencies must be turned off and the orography must be made flat. This subsection describes how these can be controlled.

Among the boundary conditions listed in the previous section, the orography $\Phi|_{\sigma=1}$ alone directly affects the dynamics part: it affects the divergence tendency through Eq. (A.2) and (A.9). All the other boundary conditions enter the prognostic equations (A.1)–(A.5) only through the physics tendencies. Therefore, effects of boundary conditions other than the orography onto the dynamical core automatically vanishes by switching the physics off.

In the SPEEDY model, physics tendencies $\left.\dfrac{d(\cdot)}{dt}\right|_{\mathbf{Phys}}$ are computed and added to the total tendencies in the subroutine called `PHYPAR()`. Therefore, switching-off of physical parametrizations can be done simply by commenting-out calls to this subroutine.

Appendix 2: Phase III

A2.1 Approach

In Phase III, I will reproduce the estimation-correction stage of Danforth et al. (2007). On each time step of the integration of the model M^{LF}, the model error statistics obtained in Phase II will be used to estimate the model error. The model error will be reduced by subtracting the estimated error from the predicted model state. The procedure will be repeated for the models with the newly implemented schemes (M^{Ncyc} and M^{RK4}) as well.

A2.2 Algorithm

The detailed algorithm for Phase III is as follows:
During the integration of M^{LF} (or M^{Ncyc}, M^{RK4}), on each time step $t = t_0$:

1. We estimate the state-dependent component of model error by regressing the current model state $x(t_0)$ onto the model error in the space spanned by the singular vectors. Namely, the anomaly of the current model state $x(t_0) - \langle x^{true} \rangle$ is expressed as a linear combination of left singular vectors u_i. The coefficients $a_i(t_0)$ can be computed by simply taking projection of $x(t_0) - \langle x^{true} \rangle$ onto u_i because of the orthonormality of u_i. Having obtained u_i for sufficiently many modes, we can now "reconstruct" the model error by performing regression in the space spanned by singular vectors:

$$\begin{aligned} a_i(t_0) &= \big(x(t_0) - \langle x^{true} \rangle\big) \cdot u_i \\ \delta x^{dep}(t_0) &= \sum_i \sigma_i a_i(t_0) v_i \\ \because \tfrac{\langle a_i(t), b_i(t) \rangle}{\langle a_i(t)^2 \rangle} &= \sigma_i \quad \text{from the property of SVD} \end{aligned}$$

   The total estimated model error for 6-hour forecast is then expressed as the sum of state-independent component (i.e. the bias) and the state-dependent component $\delta x^{dep}(t_0)$ obtained above:

$$\delta x^{tot}(t_0) = \langle \delta x(t) \rangle + \delta x^{dep}(t_0)$$

2. The single step forecast is then corrected by subtracting the model error which is scaled to fit the time stepping of the model ($2\Delta t$ for the

leapfrog, and Δt) for Runge-Kutta or Lorenz N-cycle):

$$
\begin{aligned}
M^{LF}(x(t_0)) &\leftarrow M^{LF}(x(t_0)) - \frac{2\Delta t}{6\text{hours}}\delta x^{tot}(t_0) \\
\text{or} \quad M^{Ncyc}(x(t_0)) &\leftarrow M^{Ncyc}(x(t_0)) - \frac{\Delta t}{6\text{hours}}\delta x^{tot}(t_0), \\
M^{RK4}(x(t_0)) &\leftarrow M^{RK4}(x(t_0)) - \frac{\Delta t}{6\text{hours}}\delta x^{tot}(t_0).
\end{aligned}
$$

A2.3 Implementation

The subroutines to perform error estimation and correction will be embedded to the SPEEDY model. Since the SPEEDY model is coded in Fortran 77, the subroutines will also be coded in Fortran 77.

A2.4 Validation

Again, validation of the implementation will be conducted by comparing the results for M^{LF} with those published in Danforth et al. (2007).

A2.5 Deliverables

The delivarables of this phase will be the subroutines of the SPEEDY model which performs model error estimation and correction.