U.S. DEPARTMENT OF ENERGY | Office of Science

DOE/SC-ARM-TR-253

Description of the Three-Dimensional Large-Scale Forcing Data from the 3D Constrained Variational Analysis (VARANAL3D)

S Tang
M Zhang
S Xie

August 2020

ARM

DISCLAIMER

DOE/SC-ARM-TR-253

Description of the Three-Dimensional Large-Scale Forcing Data from the 3D Constrained Variational Analysis (VARANAL3D)

S Tang, Lawrence Livermore National Laboratory (LLNL)
S Xie, LLNL
M Zhang, Stony Brook University

August 2020

Work supported by the U.S. Department of Energy,
Office of Science, Office of Biological and Environmental Research

Acronyms and Abbreviations

1D	one-dimensional
1DCVA	one-dimensional constrained variational analysis
3D	three-dimensional
3DCVA	three-dimensional constrained variational analysis
ABRFC	Arkansas-Red River Basin Forecast Center
ARM	Atmospheric Radiation Measurement
CESM	Community Earth System Model
CFSR	Climate Forecast System Reanalysis
CRM	cloud-resolving model
EBBR	energy budget Bowen ratio station
ECOR	eddy correlation flux measurement system
GOES	Geostationary Operational Environment Satellite
IDL	Interactive Data Language
JRA-55	Japanese 55-year Reanalysis
KAM	Kansas Mesonet
LES	large-eddy simulation
MC3E	Midlatitude Continental Convective Clouds Experiment
MERRA	Modern-Era Retrospective Analysis for Research and Applications
MWR	microwave radiometer
NARR	North American Regional Reanalysis
NCEP-2	National Centers for Environmental Prediction Version 2
OKM	Oklahoma Mesonet
RAP	Rapid Refresh
RRTMG	Rapid Radiative Transfer Model for general circulation models
RUC	Rapid Update Cycle
SCM	single-column model
SGP	Southern Great Plains
SIROS	Solar and Infrared Observing System
SMOS	surface meteorological observation station
TOA	top of atmosphere
VARANAL	Constrained Variational Analysis
VARANAL3D	Three-Dimensional Constrained Variational Analysis
VISST	Visible Infrared Solar-Infrared Split Window Technique

Contents

Figures

1.0 Introduction

This technical report introduces a Three-Dimensional Constrained Variational Analysis (3DCVA) (Tang and Zhang 2015) and its product of three-dimensional large-scale forcing data to drive single-column models (SCM), cloud-resolving models (CRM), and large-eddy simulation (LES) models, and to evaluate model results. The 3DCVA algorithm is an extension of the original 1D constrained variational analysis (1DCVA) (Zhang and Lin 1997, Zhang et al. 2001). The three-dimensional structure of the forcing data allows studies of spatial variation of the large-scale forcing fields and tests of physical parameterizations across scales. In the U.S. Department of Energy Atmospheric Radiation Measurement (ARM) user facility, the 3D forcing data are assigned the datastream name varanal3d. In this technical report, 3DCVA will be used to refer to the algorithm, while VARANAL3D will be used to refer to the data product.

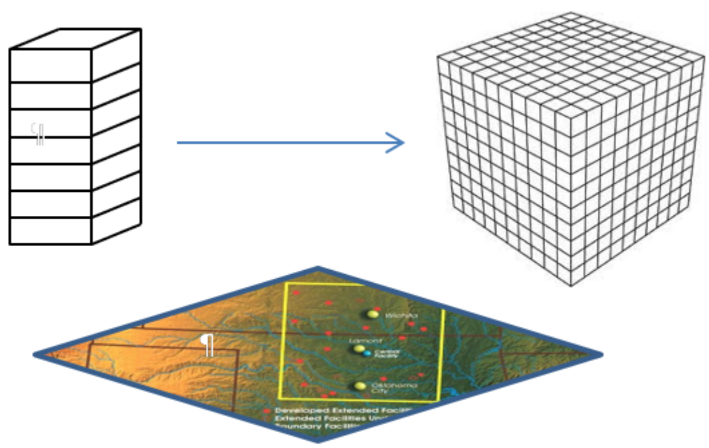

Figure 1. Schematic of changing from 1DCVA to 3DCVA.

Compared to the original 1DCVA method, the 3DCVA has the following improvements:

- The 3DCVA extends the original method from one atmospheric column into many sub-columns, within a similar size of domain. The constraint equations are satisfied in each sub-column and all sub-columns interact with each other through horizontal fluxes.
- In addition to the mass, moisture, and heat constraints used in 1DCVA, radiative constraints are also used in 3DCVA to better constrain the apparent heating and cooling at upper layers near the tropopause.
- The 3DCVA incorporates horizontal and vertical correlations in the error covariance matrix. The new error covariance matrix describes the correlation between different spatial grids more realistically.
- The 3DCVA adjusts the logarithm of water vapor mixing ratio *ln(q)* instead of water vapor mixing ratio *q* in 1DCVA to guarantee a positive definite moisture field.
- An ensemble framework is built by using multiple reanalyses/analyses data as background data to characterize data uncertainties.

2.0 Model Description

The 3DCVA algorithm adjusts the upper-level background data (*u*, *v*, *q*, *s*) within their uncertainties to satisfy a set of constraint equations in each grid box of the analysis domain. The column-integrated constraint equations are the same as those used in the 1DCVA. Following Zhang and Lin (1997), the conservation of mass, water vapor, heat, and momentum in an atmospheric column can be written as:

$$\left\langle \nabla \cdot \vec{V} \right\rangle = -\frac{1}{g}\frac{dP_s}{dt} \tag{1}$$

$$\frac{\partial \langle q \rangle}{\partial t} + \left\langle \nabla \cdot \vec{V} q \right\rangle = E_s - P_{rec} - \frac{\partial \langle q_l \rangle}{\partial t} + \frac{\omega_s q_s}{g} \tag{2}$$

$$\frac{\partial \langle s \rangle}{\partial t} + \left\langle \nabla \cdot \vec{V} s \right\rangle = R_{TOA} - R_{SRF} + L_v P_{rec} + SH + L_v \frac{\partial \langle q_l \rangle}{\partial t} + \frac{\omega_s s_s}{g} \tag{3}$$

$$\frac{\partial \left\langle \vec{V} \right\rangle}{\partial t} + \left\langle \nabla \cdot \vec{V}\vec{V} \right\rangle + f\vec{k} \times \left\langle \vec{V} \right\rangle + \nabla \langle \phi \rangle = \overrightarrow{\tau_s} \tag{4}$$

In the constraint equations above, the bracket represents vertical integration from the surface to the top of atmosphere (TOA); $s = C_p T + gz$ is the dry static energy; E_s is surface evaporation; P_{rec} is surface precipitation; L_v is the latent heat of vaporization; q_l is cloud liquid water content; R_{TOA} and R_{SRF} are net downward radiation flux at TOA and surface; *SH* is surface sensible heat flux; *f* is the Coriolis parameter; $\vec{k}$ is the unit vector in vertical direction; ϕ is the geopotential; and $\overrightarrow{\tau_s}$ is the surface wind stress. Other variables are as commonly used in meteorology. The terms related with the ω_s in Equations (2) and (3) are from the vertical integration of the three-dimensional divergence terms.

2.1 Radiative Heating Constraint Above Cloud Top

As shown in Yanai et al. (1973), the apparent heating source Q1 and apparent moisture sink Q2 can be calculated from the large-scale upper-level state variables:

$$Q_1 = \frac{\partial \bar{s}}{\partial t} + \overline{\overline{\vec{V}}} \cdot \nabla \bar{s} + \bar{\omega}\frac{\partial \bar{s}}{\partial p} = Q_{rad} + L_v (c - e) - \frac{\partial \overline{\omega' s'}}{\partial p} \tag{5}$$

$$Q_2 = -L_v \left(\frac{\partial \bar{q}}{\partial t} + \overline{\overline{\vec{V}}} \cdot \nabla \bar{q} + \bar{\omega}\frac{\partial \bar{q}}{\partial p} \right) = L_v (c - e) + L_v \frac{\partial \overline{\omega' q'}}{\partial p} \tag{6}$$

Where Q_{rad} is radiative heating; c is condensation; e is evaporation; the overbar refers to horizontal average in each grid (omitted in later equations), and the prime refers to deviation from the average. The heating and drying due to ice processes are neglected. When calculating Q1 from the left-hand side of Equation (5), we often found large spurious heating or cooling centers at upper layers near the tropopause, where no strong diabatic heating sources or sinks should exist except radiation. The large heating or cooling centers at these layers are mainly caused by amplification of the vertical velocity errors to Q1 via vertical advections, where the vertical gradient of dry static energy s is large. We therefore introduce another physical constraint to address this issue.

The apparent heating Q1 in the atmosphere includes latent heating, radiative heating, and sub-grid-scale heat transport. In the region with little sub-grid-scale heat transport and without cloud, the only heating source is radiative heating/cooling. The radiative heating rate should be equal to Q1 calculated from the large-scale dynamics. Therefore, we impose a radiative constraint:

$$Q_1 = \frac{\partial s}{\partial t} + \vec{V} \cdot \nabla s + \omega \frac{\partial s}{\partial p} = Q_{rad} \tag{7}$$

at each layer above observed cloud top or 400hPa, whichever is higher. The new constraint removes spurious large heating and cooling centers in the upper troposphere.

2.2 Adjust *ln(q)* instead of *q*

The uncertainties of atmospheric state variables are usually assumed by following Gaussian distributions. This is a reasonable assumption for *u*, *v*, and *s*, but not for *q* at upper layers. In the upper troposphere, the value of *q* is quite small, and so the adjustments based on Gaussian distribution can make *q* less than 0, which is unphysical. To solve this problem, we rewrite the variable *q* to $e^{\ln(q)}$. Equation (2) then becomes:

$$\frac{\partial \left\langle e^{Q} \right\rangle}{\partial t} + \left\langle \nabla \cdot \vec{V} e^{Q} \right\rangle = E_s - P_{rec} - \frac{\partial \left\langle q_l \right\rangle}{\partial t} + \frac{\omega_s q_s}{g}, \quad \text{where } Q = \ln q \tag{8}$$

The error of *Q* follows Gaussian distribution better than *q*, and we adjust *Q* as a new variable instead of *q*. It guarantees that *q* is definitely positive no matter how large the adjustment of *Q*.

Currently, momentum constraint is not incorporated in the 3DCVA, and the ice processes and advections of cloud hydrometeors are neglected for simplicity. The terms in the right-hand sides of the constraint equations are treated as “truth” and referred to as constraint variables. With these constraints, the background data are carefully adjusted; vertical velocity, advections, and other variables need for cloud models are calculated (Figure 2).

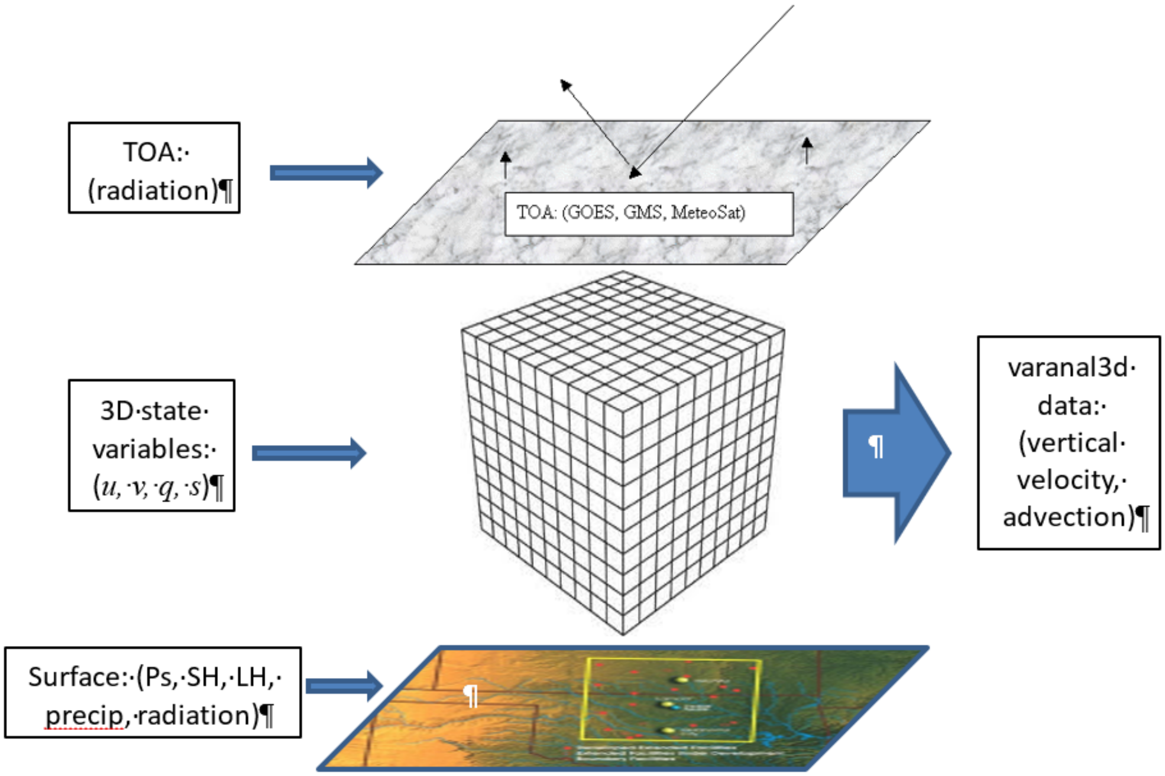

Figure 2. Schematic of 3DCVA showing the inputs of background data (3D state variables), surface and TOA constraint variables, and the output of 3D large-scale forcing data.

3.0 Numerical Algorithms

The adjustments of background data are obtained by minimizing the cost function:

$$I=\left(u-u_o\right)^T B_u^{-1}\left(u-u_o\right)+\left(v-v_o\right)^T B_v^{-1}\left(v-v_o\right)+\left(q-q_o\right)^T B_q^{-1}\left(q-q_o\right)+\left(s-s_o\right)^T B_s^{-1}\left(s-s_o\right) \quad (9)$$

where *u*, *v*, *q*, *s* are column vectors of wind, specific humidity and dry static energy for all grids in each time step. For a three-dimensional domain with I×J horizontal grids and K vertical levels, column vector $u=\left(u_{111},u_{211},...,u_{I11},u_{I21},...,u_{IJ1},...,u_{IJK}\right)^T$; *v*, *q*, and *s* are vectors in similar forms. The superscript *T* denotes the transpose of a vector; the subscript *o* denotes the initial state, and *B* represents error covariance matrix for each state variable. To better calculate the fluxes and improve the convergence of the algorithm, we use adjusted C-grid (Figure 3) for background data. In the adjusted C-grid, *u* is at the center of east/west grid faces; *v* is at the center of north/south grid faces; *s* and *q* are at the center of all four grid faces and all the constraint variables (precipitation, radiation, surface fluxes) are in grid center and represent grid averaging values.

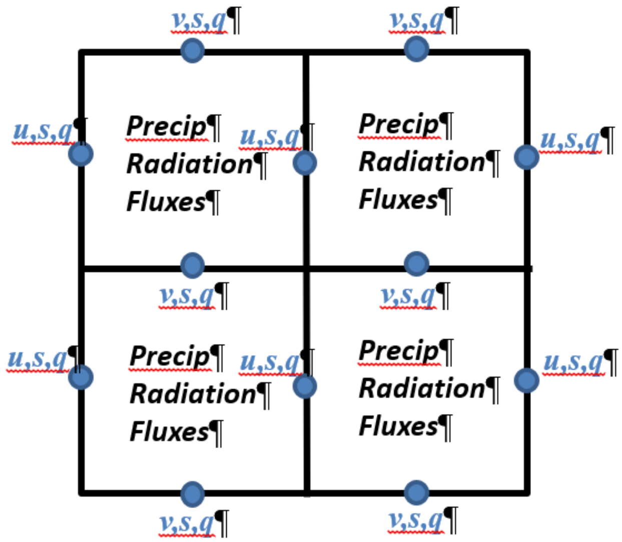

Figure 3. The adjusted C-grid used in 3DCVA.

We use the Lagrange multiplier method to minimize the cost function *I* with state variables subject to the constraint equations. The constraint equations (1, 8, 3, and 7) can be written in the residual form:

$$A_{mass} = \left\langle \nabla \cdot \vec{V} \right\rangle + \frac{1}{g}\frac{dP_s}{dt} \tag{10}$$

$$A_{water} = \frac{\partial \left\langle e^Q \right\rangle}{\partial t} + \left\langle \nabla \cdot \vec{V} e^Q \right\rangle - E_s + P_{rec} + \frac{\partial \left\langle q_l \right\rangle}{\partial t} + \frac{\omega_s q_s}{g} \tag{11}$$

$$A_{heat} = \frac{\partial \left\langle s \right\rangle}{\partial t} + \left\langle \nabla \cdot \vec{V} s \right\rangle - R_{TOA} + R_{SRF} - LP_{rec} - SH - L\frac{\partial \left\langle q_l \right\rangle}{\partial t} + \frac{\omega_s s_s}{g} \tag{12}$$

$$A_{radiation} = \frac{\partial s}{\partial t} + \vec{V} \cdot \nabla s + \omega \frac{\partial s}{\partial p} - Q_{rad} \tag{13}$$

Using Lagrange multiplier λ, we define a new cost function *J* as:

$$J(X) = I(X) + \sum_k \lambda_k A_k(X) = X^T B^{-1} X + \sum_k \lambda_k A_k(X) \tag{14}$$

where $X = (u, v, q, s)^T$ represents the vector of all background data. The error covariance matrix is

$$B = \begin{bmatrix} B_u & 0 & 0 & 0 \\ 0 & B_v & 0 & 0 \\ 0 & 0 & B_q & 0 \\ 0 & 0 & 0 & B_s \end{bmatrix} \tag{15}$$

in which we assume no correlation between different variables. The minimization of the cost function *J* becomes solving the following linearized equations:

$$\frac{\partial I(X)}{\partial x_i} + \sum_k \lambda_k \frac{\partial A_k(X)}{\partial x_i} = 0 \tag{16}$$

$$A_k(X) = 0 \tag{17}$$

where x_i represents each variable in vector *X*. If we treat all λ_k as a variable similar to x_i, and let $X^* = \{X, \lambda\} = \{x_i^*\}$, we can rewrite the linearized equations as:

$$\frac{\partial J}{\partial x_i^*} = 0 \tag{18}$$

Then, we use Newton's iteration method to solve this equation. Ignoring higher-order terms, we make the Taylor expansion of the gradient of *J* at point x_{io}^*. The previous equation then becomes:

$$\frac{\partial J}{\partial x_i^*} = \frac{\partial J}{\partial x_{io}^*} + \sum_j \frac{\partial^2 J}{\partial x_{io}^* \partial x_{jo}^*} \delta x_j^* = 0 \tag{19}$$

or

$$\sum_j \frac{\partial^2 J}{\partial x_{io}^* \partial x_{jo}^*} \delta x_j^* = -\frac{\partial J}{\partial x_{io}^*} \tag{20}$$

Error covariance matrix *B* in the cost function *J* determines how the background data are adjusted. In 3DCVA, *B* is a large, sparse matrix and could be ill-conditioned. Directly calculating the inversion of *B* is expensive and it may contain large numerical errors. We therefore first pre-condition the equation. We write Equation (20) in the following matrix form:

$$\begin{pmatrix} 2B^{-1}+\sum_k \lambda_k \dfrac{\partial A_k}{\partial x_{io}\partial x_{jo}} & \dfrac{\partial A_k}{\partial x_{jo}} \\ \left(\dfrac{\partial A_k}{\partial x_{jo}}\right)^T & 0 \end{pmatrix} \begin{pmatrix} \delta x_j \\ \delta \lambda_k \end{pmatrix} = -\begin{pmatrix} 2B^{-1}\left(x_i - x_{io}\right)+\sum_k \lambda_k \dfrac{\partial A_k}{\partial x_{io}} \\ A_k \end{pmatrix} \tag{21}$$

We then multiply the above equation by a matrix $\begin{pmatrix} B & 0 \\ 0 & I_0 \end{pmatrix}$ to cancel the inversion of *B* (I_0 is identity matrix). After some manipulation, the equation becomes

$$\begin{pmatrix} 2I_0+B\times\sum_k \lambda_k \dfrac{\partial A_k}{\partial x_{io}\partial x_{jo}} & B\times\dfrac{\partial A_k}{\partial x_{jo}} \\ \left(\dfrac{\partial A_k}{\partial x_{jo}}\right)^T & 0 \end{pmatrix} \begin{pmatrix} \delta x_j \\ \delta \lambda_k \end{pmatrix} = -\begin{pmatrix} 2\left(x_i - x_{io}\right)+B\times\sum_k \lambda_k \dfrac{\partial A_k}{\partial x_{io}} \\ A_k \end{pmatrix} \tag{22}$$

This is the linear matrix equation we solve in 3DCVA.

Theoretically, error covariance matrix B should be related to the difference between the first guess (background data) and the truth, but the truth is never known. The 1DCVA uses the time variance of the background data plus instrument and measurement uncertainty to represent the error covariance matrix, while it assumes errors at different locations and for different variables are independent. However, the time variance includes information related to the natural variability other than the data uncertainty. When we do the ensemble 3DCVA with different background data (see the next section), we use the covariance of differences among different background data in each spatial grid plus instrument and measurement uncertainty as the error covariance matrix. This error covariance matrix better reflects the possible uncertainty in the data sets. Also, we include the horizontal and vertical correlations to the neighboring grids, but still assume that there is no correlation between different variables and different time steps. The time variance is still kept as one option for the calculation of error covariance that may be used in case the ensemble background data are not available.

The Newton's iteration procedure is described as follows:

1. Set the original background data as initial conditions and the initial values of all λs as 1.0.
2. Calculate first and second derivative of *A*. ($\dfrac{\partial A_k}{\partial x_{io}}$ and $\dfrac{\partial A_k}{\partial x_{io}\partial x_{jo}}$)
3. Calculate δx_j^* by solving Equation (22).
4. Update X_o^* as $x_{jo}^{*(n+1)} = x_{jo}^{*(n)} + \delta x_j^*$, where *n* is the iteration index.
5. Repeat steps 2-4 with updated values.

6. Continue the loop until the maximum iteration numbers is reached (20 in default) or the norm of the vector $\left\{\frac{\partial J}{\partial x_{io}^*}\right\}$ is less than a threshold value (0.01 in default).

Equation (22) is a typical type of linear system $Ax = b$, where A is a large, sparse, and symmetric matrix. For a typical 3DCVA domain with ~10×10 horizontal grids, A is a ~10^4×10^4 matrix. Moreover, the matrix A is ill-conditioned and the condition number increases with the numbers of grid points, the constraints, and the inclusion of horizontal and vertical correlations. For better accuracy and efficiency, we use a LSQR function in MATLAB to solve the sparse linear matrix equation. LSQR is an iteration method to solve the least squares solution x that minimizes norm$(b - A * x)$ (Paige and Saunders 1982, Barrett et al. 1994). We tuned the normalization factors and the maximum iteration numbers of LSQR and Newton's method to achieve better convergence.

4.0 The Ensemble Framework of 3DCVA

To reduce the uncertainty of the forcing data to the input data, Tang et al. (2016) introduced an ensemble approach of 3DCVA based on different background data of the atmospheric state variables, the error covariance matrix, and the constrained variables. We applied this framework to provide an ensemble 3DCVA product to improve the data quality and allow users to investigate the impacts due to the data uncertainty.

Since Tang et al. (2016) found that background data have the largest impact on the derived 3D forcing data, the ensemble 3DCVA data include different reanalyses/analysis background data from different modeling centers. The choice of the background data may be changed for different field campaigns (see Section 6 for more details). These background data are firstly interpolated into the same temporal and spatial resolution (e.g., 0.5°×0.5° horizontal resolution, 25 hPa vertical resolution, and 3-hour time resolution), then used in 3DCVA to derive 3D forcing data separately. Instead of applying 3DCVA on the ensemble background data, the ensemble 3D forcing data are obtained by averaging the 3D forcing data analyzed from using individual background data. Both the ensemble forcing and the forcing based on individual background data are provided.

5.0 The Workflow and Code Structure for VARANAL3D

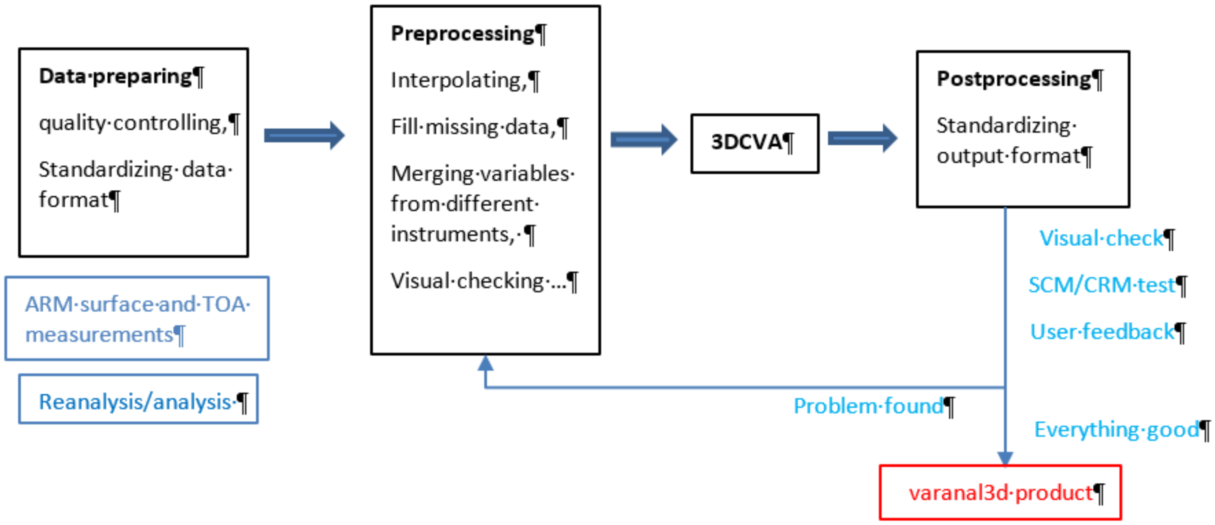

Figure 4. Illustration of the workflow.

The procedure of VARANAL3D analysis includes four steps (Figure 4), and the 3DCVA algorithm is the third step of the procedure. For the ARM SGP site, the data preparation and pre-processing codes are the same as the 1DCVA for the continuous forcing data set (Tang et al. 2019), which are written in Interactive Data Language (IDL). Some additional preparation is needed for the reanalysis data that are not archived by ARM. The 3DCVA and post-processing part are written in MATLAB, and a Rapid Radiative Transfer Model for general circulation models (RRTMG) is also used to calculate radiative heating above cloud top for the constraint Equation (13).

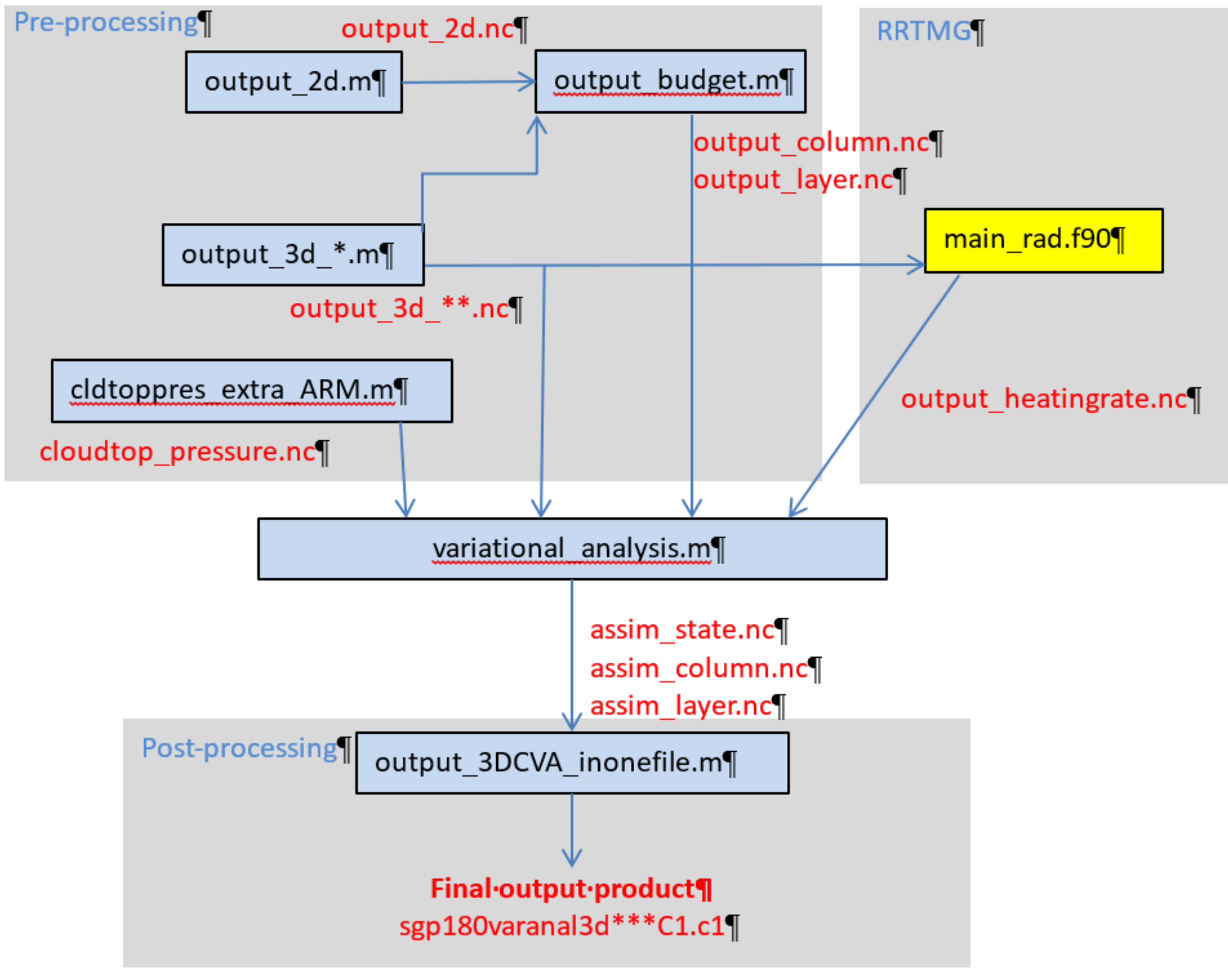

Figure 5. Illustration of the main code and input/output files of 3DCVA and post-processing steps. Code are shown in boxes while input/output files are shown in red. The yellow box represents FORTRAN code for the RRTMG model. Gray shadings indicate individual code blocks.

The code structure of 3DCVA and post-processing steps are further shown in Figure 5. *output_2d.m* reads the pre-processed 2D gridded data from VARANAL (grid_x.*) while *output_3d_*.m* reads background data from analysis/reanalysis (denoted by *). *cldtoppres_extra_ARM.m* reads the gridded cloud-top pressure from ARM Visible Infrared Solar-Infrared Split Window Technique (VISST) products. The format of these three input files can be found in Appendix 1.

In the 3DCVA code, *output_budget.m* calculates the column-integrated and 3D budgets for the original data. The *main_rad.f90* in RRTMG uses the temperature and moisture profiles in **output_3d.nc** to calculate radiative heating rate under clear-sky condition. These files along with the **cloudtop_pressure.nc** are used as input in the *variational_analysis.m* code to calculate the final adjusted upper-level fields and the new budgets. The *output_3DCVA_inonefile.m* combines all the outputs from *variational_analysis.m* into one file, which is the VARANAL3D product:

sgp180varanal3d****C1.c1.yyyymmdd.hh0000.nc

The VARANAL3D is a 3D product. To drive SCM/CRM/LESs, users can use the MATLAB code *generate_forcing_3DCVA.m* (provided along with the product) to generate 1D forcing data, with options to average in different spatial resolution. The example file head of final varanal3d product is shown in Appendix 2.

6.0 Available Data Sets

Currently, the VARANAL3D products are available for two field campaigns at SGP: the March 2000 cloud intensive operational period (IOP; Xie et al. 2005, Tang and Zhang 2015, Tang et al. 2016) and the Midlatitude Continental Convective Clouds Experiment (MC3E) (Xie et al. 2014). Ensemble products are available for both field campaigns, but the background data of the ensemble members are different.

6.1 March 2000 (0003) IOP

The March 2000 cloud IOP was conducted from 1 March to 22 March 2000 at the ARM SGP site. The surface and TOA constraints for 0003 IOP are similar to those used in VARANAL, which include:

- SMOS: Surface meteorological observation stations measuring surface precipitation, surface pressure, surface winds, temperature, and relative humidity.
- EBBR: Energy budget Bowen ratio stations measuring surface latent and sensible heat fluxes and surface broadband net radiative flux.
- ECOR: Eddy correlation flux measurement systems measuring surface latent and sensible heat fluxes.
- OKM and KAM: Oklahoma and Kansas Mesonet stations measuring surface precipitation, pressure, winds, and temperature.
- MWR: Microwave radiometer stations measuring the column precipitable water and total cloud liquid water.
- SIROS: Solar and Infrared Observing Systems measuring broadband longwave and shortwave radiative fluxes.
- GOES: the Geostationary Operational Environment Satellite measuring radiative fluxes at TOA.
- ABRFC: the 4-km-resolution gridded precipitation products from Arkansas-Red Basin River Forecast Center based on WSD-88 rain radar and gauge measurements.

The six reanalysis/analysis data used as background data are Rapid Update Cycle (RUC), ERA-Interim, Climate Forecast System Reanalysis (CFSR), North American Regional Reanalysis (NARR), Modern-Era Retrospective Analysis for Research and Applications (MERRA), and Japanese 55-year Reanalysis (JRA-55). Therefore, there are seven VARANAL3D products for 0003 IOP:

sgp180varanal3dcfsrC1.c1.20000301.150000.nc

sgp180varanal3densembleC1.c1.20000301.150000.nc

sgp180varanal3dinterimC1.c1.20000301.150000.nc

sgp180varanal3djra55C1.c1.20000301.150000.nc

sgp180varanal3dmerraC1.c1.20000301.150000.nc *sgp180varanal3dnarrC1.c1.20000301.150000.nc*

sgp180varanal3drucC1.c1.20000301.150000.nc

6.2 MC3E

MC3E was conducted during April to June 2011 near the ARM SGP site. The analysis covers the period from 00Z April 22–21Z June 6, 2011. The surface and TOA constraints for MC3E are the same as those for the 0003 IOP, but the six background fields are now Rapid Refresh (RAP), ERA5, National Centers for Environmental Prediction Version 2 (NCEP-2), NARR, MERRA-2, and JRA-55. Therefore, the VARANAL3D products for MC3E are:

sgp180varanal3densembleC1.c1.20110422.000000.nc

sgp180varanal3dera5C1.c1.20110422.000000.nc

sgp180varanal3djra55C1.c1.20110422.000000.nc

sgp180varanal3dmerraC1.c1.20110422.000000.nc

sgp180varanal3dnarrC1.c1.20110422.000000.nc

sgp180varanal3dncepC1.c1.20110422.000000.nc

sgp180varanal3drucC1.c1.20110422.000000.nc

7.0 Data Information and Contact

The VARANAL3D products are available to the community from the ARM Data Center (http://www.archive.arm.gov/discovery/). The data include files using individual reanalysis/analysis data as background fields, and the ensemble of them. A MATLAB code is also provided to convert the VARANAL3D product to the forcing data for driving SCM of Community Earth System Model (CESM) (SCAM5), with options to average in different spatial resolution. More information about the 3DCVA algorithm can be found in Tang and Zhang (2015) and Tang et al. (2016, 2017).

For questions or comments, please contact:

Shuaiqi Tang
Atmospheric, Earth, and Energy Division (L-103)
Lawrence Livermore National Laboratory
Livermore, California 94550
tang32@llnl.gov

Shaocheng Xie
Atmospheric, Earth, and Energy Division (L-103)
Lawrence Livermore National Laboratory
Livermore, California 94550
xie2@llnl.gov

8.0 References

Barrett, R, M Berry, TF Chan, J Demmel, J Donato, J Dongarra, V Eijkhout, R Pozo, C Romine, and H Van der Vorst. 1994. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. Society for Industrial and Applied Mathematics, Philadelphia, Pennsylvania.

Paige, CC, and MA Saunders. 1982. "LSQR: An Algorithm for Sparse Linear Equations and Sparse Least Squares." *ACM Transactions on Mathematical Software* 8(1): 43−71, https://doi.org/10.1145/355984.355989

Tang, S, and M Zhang. 2015. "Three-dimensional constrained variational analysis: Approach and application to analysis of atmospheric diabatic heating and derivative fields during an ARM SGP intensive observational period." *Journal of Geophysical Research − Atmospheres* 120(15): 7283−7299, https://doi.org/10.1002/2015JD023621

Tang, S, M Zhang, and S Xie. 2016. "An ensemble constrained variational analysis of atmospheric forcing data and its application to evaluate clouds in CAM5." *Journal of Geophysical Research − Atmospheres* 121(1): 33−48, https://doi.org/10.1002/2015JD024167

Tang, S, M Zhang, and S Xie. 2017. "Investigating the dependence of SCM simulated precipitation and clouds on the spatial scale of large-scale forcing at SGP." *Journal of Geophysical Research − Atmospheres* 122(16): 8724−8738, https://doi.org/10.1002/2017JD026565

Tang, S, C Tao, S Xie, and M Zhang. 2019. Description of the ARM Large-Scale Forcing Data from the Constrained Variational Analysis (VARANAL) Version 2. U.S. Department of Energy. DOE/SC-ARM-TR-222.

Xie, S, Y Zhang, SE Giangrande, MP Jensen, R McCoy, and M Zhang. 2014. "Interactions between Cumulus Convection and Its Environment as Revealed by the MC3E Sounding Array." *Journal of Geophysical Research − Atmospheres*, 2014JD022011, https://doi.org/10.1002/2014JD022011

Xie, S, M Zhang, M Branson, RT Cederwall, AD Del Genio, ZA Eitzen, SJ Ghan, SF Iacobellis, KL Johnson, M Khairoutdinov, SA Klein, SK Krueger, W Lin, U Lohmann, MA Miller, DA Randall, RCJ Somerville, YC Sud, GK Walker, A Wolf, X Wu, K-M X, JJ Yo, G Zhang, and J Zhang. 2005. "Simulations of midlatitude frontal clouds by single-column and cloud-resolving models during the Atmospheric Radiation Measurement March 2000 cloud intensive operational period." *Journal of Geophysical Research − Atmospheres* 110(D15): D15S03, https://doi.org/10.1029/2004JD005119

Yanai, M, S Esbensen, and J-H Chu. 1973. "Determination of Bulk Properties of Tropical Cloud Clusters from Large-Scale Heat and Moisture Budgets." *Journal of the Atmospheric Sciences* 30(4): 611−627, https://doi.org/10.1175/1520-0469(1973)030<0611:DOBPOT>2.0.CO;2

Zhang, M, and J Lin. 1997. "Constrained Variational Analysis of Sounding Data Based on Column-Integrated Budgets of Mass, Heat, Moisture, and Momentum: Approach and Application to ARM Measurements." *Journal of the Atmospheric Sciences* 54(11): 1503−1524, https://doi.org/10.1175/1520-0469(1997)054<1503:CVAOSD>2.0.CO;2

Zhang, M, J Lin, RT Cederwall, JJ Yio, and SC Xie. 2001. "Objective Analysis of ARM IOP Data: Method and Sensitivity." *Monthly Weather Review* 129(2): 295−311, https://doi.org/10.1175/1520-0493(2001)129<0295:OAOAID>2.0.CO;2

Appendix A

Examples of VARANAL3D Input File Heads

output_2d.nc

```
netcdf output_2d {
dimensions:
      lon = 10 ;
      lat = 9 ;
      time = 168 ;
variables:
      double time(time) ;
            time:long_name = "Time" ;
            time:units = "minutes since 2000-03-01 15:00" ;
      double lon(lon) ;
            lon:long_name = "longitudes" ;
            lon:units = "degrees_east" ;
      double lat(lat) ;
            lat:long_name = "latitudes" ;
            lat:units = "degrees_north" ;
      double sfc_pres(time, lat, lon) ;
            sfc_pres:long_name = "surface pressure" ;
            sfc_pres:units = "Pa" ;
      double precip(time, lat, lon) ;
            precip:long_name = "precip * Lv0/Cpd" ;
            precip:units = "K/s" ;
      double lprecip(time, lat, lon) ;
            lprecip:long_name = "precip *Lv/Cpd" ;
            lprecip:units = "K/s" ;
      double evapor(time, lat, lon) ;
            evapor:long_name = "Evaporation * Lv0/Cpd" ;
            evapor:units = "K/s" ;
      double shf(time, lat, lon) ;
            shf:long_name = "Sensible heating rate" ;
            shf:units = "K/s" ;
      double qc(time, lat, lon) ;
            qc:long_name = "cloud water * Lv0/Cpd" ;
            qc:units = "K" ;
      double radiationt(time, lat, lon) ;
            radiationt:long_name = "Radiative heating-TOA" ;
            radiationt:units = "K/s" ;
```

```
        double radiationb(time, lat, lon) ;
                radiationb:long_name = "Radiative heating-SFC" ;
                radiationb:units = "K/s" ;
        double radiation(time, lat, lon) ;
                radiation:long_name = "Radiative heating-COL" ;
                radiation:units = "K/s" ;
        double taox(time, lat, lon) ;
                taox:long_name = "X-shearing stress" ;
                taox:units = "N" ;
        double taoy(time, lat, lon) ;
                taoy:long_name = "Y-shearing stress" ;
                taoy:units = "N" ;
}
```

output_3d.nc

```
netcdf output_3d {
dimensions:
        lon = 10 ;
        lat = 9 ;
        lev = 27 ;
        time = 168 ;
variables:
        double lon(lon) ;
                lon:long_name = "longitude" ;
                lon:units = "degrees_east" ;
        double lat(lat) ;
                lat:long_name = "latitude" ;
                lat:units = "degrees_north" ;
        double lev(lev) ;
                lev:long_name = "level" ;
                lev:units = "millibar" ;
        double time(time) ;
                time:long_name = "Time" ;
                time:units = "minutes since 2000-03-01 15:00" ;
        double kb(time, lat, lon) ;
                kb:long_name = "the lowest level index" ;
                kb:units = "-" ;
        double usfc(time, lat, lon) ;
                usfc:long_name = "surface U wind" ;
                usfc:units = "m/s" ;
        double vsfc(time, lat, lon) ;
                vsfc:long_name = "surface V wind" ;
                vsfc:units = "m/s" ;
        double tsfc(time, lat, lon) ;
                tsfc:long_name = "surface temperature" ;
                tsfc:units = "K" ;
        double psfc(time, lat, lon) ;
                psfc:long_name = "surface pressure" ;
                psfc:units = "Pa" ;
        double qsfc(time, lat, lon) ;
                qsfc:long_name = "surface specific humidity" ;
```

```
            qsfc:units = "kg/kg" ;
      double zsfc(time, lat, lon) ;
            zsfc:long_name = "surface geopotential height" ;
            zsfc:units = "gpm" ;
      double ssfc(time, lat, lon) ;
            ssfc:long_name = "surface static energy (/Cpd)" ;
            ssfc:units = "K" ;
      double u(time, lev, lat, lon) ;
            u:long_name = "U wind" ;
            u:units = "m/s" ;
      double v(time, lev, lat, lon) ;
            v:long_name = "V wind" ;
            v:units = "m/s" ;
      double q(time, lev, lat, lon) ;
            q:long_name = "specific humidity" ;
            q:units = "kg/kg" ;
      double s(time, lev, lat, lon) ;
            s:long_name = "static energy (/Cpd)" ;
            s:units = "K" ;
      double T(time, lev, lat, lon) ;
            T:long_name = "temperature" ;
            T:units = "K" ;
      double w(time, lev, lat, lon) ;
            w:long_name = "vertical velocity" ;
            w:units = "Pa/s" ;
      double z(time, lev, lat, lon) ;
            z:long_name = "geopotential height" ;
            z:units = "gpm" ;
}
```

cloudtop_pressure.nc

```
netcdf cloudtop_pressure {
dimensions:
      lat = 9 ;
      lon = 10 ;
      time = 168 ;
variables:
      double time(time) ;
            time:units = "minutes since 2000-03-01 15:00" ;
      double lat(lat) ;
            lat:units = "degrees_north" ;
      double lon(lon) ;
            lon:units = "degrees_east" ;
      float cldtoppres(time, lat, lon) ;
            cldtoppres:units = "hPa" ;
            cldtoppres:long_name = "highest cloud top pressure" ;
            cldtoppres:missing_value = 999.9 ;
}
```

Appendix B

Examples of VARANAL3D Output File Heads

```
netcdf sgp180varanal3densembleC1.c1.20000301.150000 {
dimensions:
      time = 168 ;
      lat = 9 ;
      lon = 10 ;
      lon_stag = 11 ;
      lat_stag = 10 ;
      lev = 37 ;
variables:
      int base_time ;
            base_time:long_name = "Base time in Epoch" ;
            base_time:units = "seconds since 1970-1-1 0:00:00 0:00" ;
            base_time:ancillary_variables = "time_offset" ;
      double time_offset(time) ;
            time_offset:long_name = "Time offset from base_time" ;
            time_offset:units = "seconds since 2000-03-01 15:00" ;
            time_offset:ancillary_variables = "base_time" ;
      double time(time) ;
            time:long_name = "Time offset from midnight" ;
            time:units = "seconds since 2000-03-01 00:00:00" ;
            time:standard_name = "time" ;
      float u(time, lev, lat, lon) ;
            u:long_name = "U-wind at grid center" ;
            u:standard_name = "eastward_wind" ;
            u:units = "m/s" ;
            u:missing_value = -9999. ;
      float v(time, lev, lat, lon) ;
            v:long_name = "V-wind at grid center" ;
            v:standard_name = "northward_wind" ;
            v:units = "m/s" ;
            v:missing_value = -9999. ;
      float q(time, lev, lat, lon) ;
            q:long_name = "water vapor mixing ratio at grid center" ;
            q:standard_name = "humidity_mixing_ratio" ;
            q:units = "kg/kg" ;
            q:missing_value = -9999. ;
      float s(time, lev, lat, lon) ;
```

```
            s:long_name = "dry static energy/Cpd (T+g*z/Cpd) at grid
center" ;
            s:standard_name = "N/A" ;
            s:units = "K" ;
            s:missing_value = -9999. ;
      float z(time, lev, lat, lon) ;
            z:long_name = "geopotential height" ;
            z:standard_name = "geopotential_height" ;
            z:units = "gpm" ;
            z:missing_value = -9999. ;
      float u_xstag(time, lev, lat, lon_stag) ;
            u_xstag:long_name = "U-wind at W-E boundary" ;
            u_xstag:standard_name = "eastward_wind" ;
            u_xstag:units = "m/s" ;
            u_xstag:missing_value = -9999. ;
      float q_xstag(time, lev, lat, lon_stag) ;
            q_xstag:long_name = "water vapor mixing ratio at W-E
boundary" ;
            q_xstag:standard_name = "humidity_mixing_ratio" ;
            q_xstag:units = "kg/kg" ;
            q_xstag:missing_value = -9999. ;
      float s_xstag(time, lev, lat, lon_stag) ;
            s_xstag:long_name = "dry static energy/Cpd (T+g*z/Cpd) at
W-E boundary" ;
            s_xstag:standard_name = "N/A" ;
            s_xstag:units = "K" ;
            s_xstag:missing_value = -9999. ;
      float v_ystag(time, lev, lat_stag, lon) ;
            v_ystag:long_name = "V-wind at N-S boundary" ;
            v_ystag:standard_name = "northward_wind" ;
            v_ystag:units = "m/s" ;
            v_ystag:missing_value = -9999. ;
      float q_ystag(time, lev, lat_stag, lon) ;
            q_ystag:long_name = "water vapor mixing ratio at N-S
boundary" ;
            q_ystag:standard_name = "humidity_mixing_ratio" ;
            q_ystag:units = "kg/kg" ;
            q_ystag:missing_value = -9999. ;
      float s_ystag(time, lev, lat_stag, lon) ;
            s_ystag:long_name = "dry static energy/Cpd (T+g*z/Cpd) at
N-S boundary" ;
            s_ystag:standard_name = "N/A" ;
            s_ystag:units = "K" ;
            s_ystag:missing_value = -9999. ;
      float div(time, lev, lat, lon) ;
            div:long_name = "Divergence of wind" ;
            div:standard_name = "divergence_of_wind" ;
            div:units = "1/s" ;
            div:missing_value = -9999. ;
      float m_domega_dp(time, lev, lat, lon) ;
            m_domega_dp:long_name = "Negative domega/dp" ;
            m_domega_dp:standard_name = "N/A" ;
```

```
            m_domega_dp:units = "1/s" ;
            m_domega_dp:missing_value = -9999. ;
      float omega(time, lev, lat, lon) ;
            omega:long_name = "Pressure vertical velocity" ;
            omega:standard_name = "lagrangian_tendency_of_air_pressure"
;
            omega:units = "Pa/s" ;
            omega:missing_value = -9999. ;
      float dq_dt(time, lev, lat, lon) ;
            dq_dt:long_name = "Time change of moisture" ;
            dq_dt:standard_name = "N/A" ;
            dq_dt:units = "kg/kg/s" ;
            dq_dt:missing_value = -9999. ;
      float h_advq(time, lev, lat, lon) ;
            h_advq:long_name = "horizontal advection of q" ;
            h_advq:standard_name = "N/A" ;
            h_advq:units = "kg/kg/s" ;
            h_advq:missing_value = -9999. ;
      float v_advq(time, lev, lat, lon) ;
            v_advq:long_name = "vertical advection of q" ;
            v_advq:standard_name = "N/A" ;
            v_advq:units = "kg/kg/s" ;
            v_advq:missing_value = -9999. ;
      float Q2(time, lev, lat, lon) ;
            Q2:long_name = "Apparent moisture sink (/cpd) from Yanai et
al., (1973)" ;
            Q2:standard_name = "N/A" ;
            Q2:units = "K/s" ;
            Q2:missing_value = -9999. ;
      float wq(time, lev, lat, lon) ;
            wq:long_name = "vertical pressure velocity multiplied by
moisture (*L/cpd)" ;
            wq:standard_name = "N/A" ;
            wq:units = "Pa*K/s" ;
            wq:missing_value = -9999. ;
      float ds_dt(time, lev, lat, lon) ;
            ds_dt:long_name = "time tendency of dry static energy
(/cpd)" ;
            ds_dt:standard_name = "N/A" ;
            ds_dt:units = "K/s" ;
            ds_dt:missing_value = -9999. ;
      float h_advs(time, lev, lat, lon) ;
            h_advs:long_name = "horizontal advection of s (/cpd)" ;
            h_advs:standard_name = "N/A" ;
            h_advs:units = "K/s" ;
            h_advs:missing_value = -9999. ;
      float v_advs(time, lev, lat, lon) ;
            v_advs:long_name = "vertical advection of s (/cpd)" ;
            v_advs:standard_name = "N/A" ;
            v_advs:units = "K/s" ;
            v_advs:missing_value = -9999. ;
      float Q1(time, lev, lat, lon) ;
```

```
            Q1:long_name = "Apparent heat source (/cpd) from Yanai et
al., (1973)" ;
            Q1:standard_name = "N/A" ;
            Q1:units = "K/s" ;
            Q1:missing_value = -9999. ;
      float ws(time, lev, lat, lon) ;
            ws:long_name = "vertical pressure velocity multiplied by
dry static energy (/cpd)" ;
            ws:standard_name = "N/A" ;
            ws:units = "Pa*K/s" ;
            ws:missing_value = -9999. ;
      float T(time, lev, lat, lon) ;
            T:long_name = "Temperature at grid center" ;
            T:standard_name = "air_temperature" ;
            T:units = "K" ;
            T:missing_value = -9999. ;
      float dT_dt(time, lev, lat, lon) ;
            dT_dt:long_name = "time tendency of temperature" ;
            dT_dt:standard_name = "N/A" ;
            dT_dt:units = "K/s" ;
            dT_dt:missing_value = -9999. ;
      float h_advT(time, lev, lat, lon) ;
            h_advT:long_name = "horizontal advection of T" ;
            h_advT:standard_name = "N/A" ;
            h_advT:units = "K/s" ;
            h_advT:missing_value = -9999. ;
      float v_advT(time, lev, lat, lon) ;
            v_advT:long_name = "vertical advection of T" ;
            v_advT:standard_name = "N/A" ;
            v_advT:units = "K/s" ;
            v_advT:missing_value = -9999. ;
      float RH(time, lev, lat, lon) ;
            RH:long_name = "relative humidity" ;
            RH:standard_name = "relative_humidity" ;
            RH:units = "%" ;
            RH:missing_value = -9999. ;
      float zs(time, lat, lon) ;
            zs:long_name = "surface geopotential height" ;
            zs:standard_name =
"surface_height_above_geopotential_datum" ;
            zs:units = "m" ;
            zs:missing_value = -9999. ;
      float lh(time, lat, lon) ;
            lh:long_name = "Downward surface latent heat" ;
            lh:standard_name = "surface_downward_latent_heat_flux" ;
            lh:units = "W/m^2" ;
            lh:missing_value = -9999. ;
      float sh(time, lat, lon) ;
            sh:long_name = "Downward surface sensible heat" ;
            sh:standard_name = "surface_downward_sensible_heat_flux" ;
            sh:units = "W/m^2" ;
            sh:missing_value = -9999. ;
```

```
      float ps(time, lat, lon) ;
            ps:long_name = "surface pressure" ;
            ps:standard_name = "air_pressure" ;
            ps:units = "hPa" ;
            ps:missing_value = -9999. ;
      float Ts(time, lat, lon) ;
            Ts:long_name = "surface temperature" ;
            Ts:standard_name = "air_temperature" ;
            Ts:units = "degC" ;
            Ts:missing_value = -9999. ;
      float rhs(time, lat, lon) ;
            rhs:long_name = "surface relative humidity" ;
            rhs:standard_name = "relative_humidity" ;
            rhs:units = "%" ;
            rhs:missing_value = -9999. ;
      float us(time, lat, lon) ;
            us:long_name = "surface U wind" ;
            us:standard_name = "eastward_wind" ;
            us:units = "m/s" ;
            us:missing_value = -9999. ;
      float vs(time, lat, lon) ;
            vs:long_name = "surface V wind" ;
            vs:standard_name = "northward_wind" ;
            vs:units = "m/s" ;
            vs:missing_value = -9999. ;
      float srflwdn(time, lat, lon) ;
            srflwdn:long_name = "surface downward longwave flux" ;
            srflwdn:standard_name =
"surface_downwelling_longwave_flux_in_air" ;
            srflwdn:units = "W/m^2" ;
            srflwdn:missing_value = -9999. ;
      float srflwup(time, lat, lon) ;
            srflwup:long_name = "surface upward longwave flux" ;
            srflwup:standard_name =
"surface_upwelling_longwave_flux_in_air" ;
            srflwup:units = "W/m^2" ;
            srflwup:missing_value = -9999. ;
      float srfswdn(time, lat, lon) ;
            srfswdn:long_name = "surface downward shortwave flux" ;
            srfswdn:standard_name =
"surface_downwelling_shortwave_flux_in_air" ;
            srfswdn:units = "W/m^2" ;
            srfswdn:missing_value = -9999. ;
      float srfswup(time, lat, lon) ;
            srfswup:long_name = "surface upward shortwave flux" ;
            srfswup:standard_name =
"surface_upwelling_shortwave_flux_in_air" ;
            srfswup:units = "W/m^2" ;
            srfswup:missing_value = -9999. ;
      float liq(time, lat, lon) ;
            liq:long_name = "liquid water path" ;
            liq:standard_name = "N/A" ;
```

```
            liq:units = "cm" ;
            liq:missing_value = -9999. ;
      float vap(time, lat, lon) ;
            vap:long_name = "column precipitable water" ;
            vap:standard_name =
"lwe_thickness_of_atmosphere_mass_content_of_water_vapor" ;
            vap:units = "cm" ;
            vap:missing_value = -9999. ;
      float lwt(time, lat, lon) ;
            lwt:long_name = "TOA net longwave flux" ;
            lwt:standard_name = "toa_net_upward_longwave_flux" ;
            lwt:units = "W/m^2" ;
            lwt:missing_value = -9999. ;
      float swt(time, lat, lon) ;
            swt:long_name = "TOA net shortwave flux" ;
            swt:standard_name = "toa_net_downward_shortwave_flux" ;
            swt:units = "W/m^2" ;
            swt:missing_value = -9999. ;
      float lwc(time, lat, lon) ;
            lwc:long_name = "TOA clearsky longwave flux" ;
            lwc:standard_name =
"toa_net_upward_longwave_flux_assuming_clear_sky" ;
            lwc:units = "W/m^2" ;
            lwc:missing_value = -9999. ;
      float swc(time, lat, lon) ;
            swc:long_name = "TOA clearsky shortwave flux" ;
            swc:standard_name =
"toa_net_downward_shortwave_flux_assuming_clear_sky" ;
            swc:units = "W/m^2" ;
            swc:missing_value = -9999. ;
      float ins(time, lat, lon) ;
            ins:long_name = "TOA insolation" ;
            ins:standard_name = "toa_incoming_shortwave_flux" ;
            ins:units = "W/m^2" ;
            ins:missing_value = -9999. ;
      float clddz(time, lat, lon) ;
            clddz:long_name = "cloud depth" ;
            clddz:standard_name = "N/A" ;
            clddz:units = "km" ;
            clddz:missing_value = -9999. ;
      float cldtz(time, lat, lon) ;
            cldtz:long_name = "cloud top height" ;
            cldtz:standard_name = "height_at_cloud_top" ;
            cldtz:units = "km" ;
            cldtz:missing_value = -9999. ;
      float cldhgh(time, lat, lon) ;
            cldhgh:long_name = "high cloud amount" ;
            cldhgh:standard_name = "high_type_cloud_area_fraction" ;
            cldhgh:units = "%" ;
            cldhgh:missing_value = -9999. ;
      float cldmid(time, lat, lon) ;
            cldmid:long_name = "middle cloud amount" ;
```

```
            cldmid:standard_name = "medium_type_cloud_area_fraction" ;
            cldmid:units = "%" ;
            cldmid:missing_value = -9999. ;
      float cldlow(time, lat, lon) ;
            cldlow:long_name = "low cloud amount" ;
            cldlow:standard_name = "low_type_cloud_area_fraction" ;
            cldlow:units = "%" ;
            cldlow:missing_value = -9999. ;
      float cldtot(time, lat, lon) ;
            cldtot:long_name = "total cloud amount" ;
            cldtot:standard_name = "cloud_area_fraction" ;
            cldtot:units = "%" ;
            cldtot:missing_value = -9999. ;
      float rsd_mass(time, lat, lon) ;
            rsd_mass:long_name = "residual of mass" ;
            rsd_mass:standard_name = "N/A" ;
            rsd_mass:units = "kg/s/m^2" ;
            rsd_mass:missing_value = -9999. ;
      float m_Dps_dt(time, lat, lon) ;
            m_Dps_dt:long_name = "Negative time difference of Ps" ;
            m_Dps_dt:standard_name = "N/A" ;
            m_Dps_dt:units = "kg/s/m^2" ;
            m_Dps_dt:missing_value = -9999. ;
      float m_sdivuv(time, lat, lon) ;
            m_sdivuv:long_name = "Negative column divergence of wind" ;
            m_sdivuv:standard_name = "N/A" ;
            m_sdivuv:units = "kg/s/m^2" ;
            m_sdivuv:missing_value = -9999. ;
      float rsd_water(time, lat, lon) ;
            rsd_water:long_name = "residual of water vapor" ;
            rsd_water:standard_name = "N/A" ;
            rsd_water:units = "kg*K/s/m^2" ;
            rsd_water:missing_value = -9999. ;
      float m_dsr_dt(time, lat, lon) ;
            m_dsr_dt:long_name = "Negative time difference of vapor" ;
            m_dsr_dt:standard_name = "N/A" ;
            m_dsr_dt:units = "kg*K/s/m^2" ;
            m_dsr_dt:missing_value = -9999. ;
      float m_sdivuvr(time, lat, lon) ;
            m_sdivuvr:long_name = "Negative column divergence of
water*wind" ;
            m_sdivuvr:standard_name = "N/A" ;
            m_sdivuvr:units = "kg*K/s/m^2" ;
            m_sdivuvr:missing_value = -9999. ;
      float precip(time, lat, lon) ;
            precip:long_name = "Surface precipitation" ;
            precip:standard_name = "N/A" ;
            precip:units = "mm/hr" ;
            precip:missing_value = -9999. ;
      float evapor(time, lat, lon) ;
            evapor:long_name = "Surface evaporation" ;
            evapor:standard_name = "N/A" ;
```

```
           evapor:units = "mm/hr" ;
           evapor:missing_value = -9999. ;
      float m_dsrldt(time, lat, lon) ;
           m_dsrldt:long_name = "Negative time difference of cloud
water" ;
           m_dsrldt:standard_name = "N/A" ;
           m_dsrldt:units = "kg*K/s/m^2" ;
           m_dsrldt:missing_value = -9999. ;
      float rsd_heat(time, lat, lon) ;
           rsd_heat:long_name = "redisual of heat" ;
           rsd_heat:standard_name = "N/A" ;
           rsd_heat:units = "kg*K/s/m^2" ;
           rsd_heat:missing_value = -9999. ;
      float m_dsh_dt(time, lat, lon) ;
           m_dsh_dt:long_name = "Negative time difference of heat" ;
           m_dsh_dt:standard_name = "N/A" ;
           m_dsh_dt:units = "kg*K/s/m^2" ;
           m_dsh_dt:missing_value = -9999. ;
      float m_sdivuvh(time, lat, lon) ;
           m_sdivuvh:long_name = "Negative column divergence of
heat*wind" ;
           m_sdivuvh:standard_name = "N/A" ;
           m_sdivuvh:units = "kg*K/s/m^2" ;
           m_sdivuvh:missing_value = -9999. ;
      float radiation(time, lat, lon) ;
           radiation:long_name = "column radiative heating" ;
           radiation:standard_name = "N/A" ;
           radiation:units = "kg*K/s/m^2" ;
           radiation:missing_value = -9999. ;
      float Lprecip(time, lat, lon) ;
           Lprecip:long_name = "latent heat by precipitation" ;
           Lprecip:standard_name = "N/A" ;
           Lprecip:units = "kg*K/s/m^2" ;
           Lprecip:missing_value = -9999. ;
      float shf(time, lat, lon) ;
           shf:long_name = "surface sensible heat (plus cloud water
latent heat)" ;
           shf:standard_name = "N/A" ;
           shf:units = "kg*K/s/m^2" ;
           shf:missing_value = -9999. ;
      float KB(time, lat, lon) ;
           KB:long_name = "bottom level index" ;
           KB:standard_name = "N/A" ;
           KB:units = "1" ;
           KB:missing_value = -9999. ;
      float KT(time, lat, lon) ;
           KT:long_name = "tropopause level index" ;
           KT:standard_name = "N/A" ;
           KT:units = "1" ;
           KT:missing_value = -9999. ;
      double lon_stag(lon_stag) ;
           lon_stag:long_name = "longitude at W-E boundary" ;
```

```
            lon_stag:units = "degrees_east" ;
            lon_stag:standard_name = "longitude" ;
      double lat_stag(lat_stag) ;
            lat_stag:long_name = "latitude at N-S boundary" ;
            lat_stag:units = "degrees_north" ;
            lat_stag:standard_name = "latitude" ;
      double lon(lon) ;
            lon:long_name = "East longitude" ;
            lon:units = "degree_E" ;
            lon:standard_name = "longitude" ;
            lon:valid_min = -180. ;
            lon:valid_max = 180. ;
      double lat(lat) ;
            lat:long_name = "North latitude" ;
            lat:units = "degree_N" ;
            lat:standard_name = "latitude" ;
            lat:valid_min = -90. ;
            lat:valid_max = 90. ;
      double lev(lev) ;
            lev:long_name = "levels" ;
            lev:units = "millibar" ;
      double alt ;
            alt:long_name = "Altitude above mean sea level" ;
            alt:units = "m" ;
            alt:standard_name = "altitude" ;

// global attributes:
            :title = "3D Constrained Variational Analysis (3DCVA)
ensemble-based v1.1: March2000" ;
            :author = "Shuaiqi Tang: tang32@llnl.gov" ;
            :Conventions = "ARM-1.2" ;
            :command_line = "output_3DCVA_inonefile.m" ;
            :datastream = "NULL" ;
            :dod_version = "1.1" ;
            :process_version = "1.1" ;
            :data_level = "c1" ;
            :doi = "10.5439/1491785" ;
            :platform_id = "NULL" ;
            :site_id = "SGP" ;
            :facility_id = "NULL" ;
            :references = "Tang and Zhang (2015).
https://doi.org/10.1002/2015JD023621" ;
            :contacts = "Shuaiqi Tang (tang32@llnl.gov); Shaocheng Xie
(xie2@llnl.gov)" ;
            :history = "NULL" ;
}
```

ARM

www.arm.gov

U.S. DEPARTMENT OF
ENERGY

Office of Science