Laboratory for Aviation
and the Environment
Massachusetts Institute of Technology

18.337 Project: J-Zephyr
A new chemistry-transport model
Final report

Author:
Sebastian D. Eastham
MIT AeroAstro

Project Timeline:
Proposal: 2013-10-16
Delivery: 2013-12-16

1 Introduction

The main objective was to make the first steps in building a chemistry transport model (CTM) in Julia, exploiting the parallel programming capabilities inherent to this new language. Appendix A.1 contains the original proposal.

2 Overview

Briefly, this draft of the new CTM – named 'J-Zephyr' – is intended to contain the basic components required to model global advective transport. However, from the perspective of a parallel programming class, the central aim was to use this as a learning experience in terms of familiarization with the unique challenges associated with parallel programming such as domain decomposition, communication overhead and non-serial debugging.

3 Code

3.1 Structure

The code itself was separated into a small number of independent files. In a sample run, the code would execute as follows:

1. An input file, the path of which is the sole direct input to the function, is read in. This input file has specific formatting and indicates data such as the path to the meteorological data, the output directory, the different tracers to be modeled and so on. This data is stored in a new data type, `JZOptType`.

2. The code goes through an initialization stage. This is done in the following order:

   (a) If present, a machine file is processed to determine the number of machines and available cores. Since the code did not pass the debug stage, this functionality was not implemented; currently the machine file is simply replaced with the number of workers to be set up on the host machine. However, the code is written to leverage distributed arrays without assumptions about the physical location of each worker, so this modification should be simple

   (b) Time data is initialized and stored in a new data type, `JZTimeType`. As is the case with much of the initialization process, a custom constructor was written which could directly initialize the relevant variables given only the `JZOptType` structure of simulation options

   (c) Similarly, grid data such as cell edge coordinates, polar cap dimensions and so on are initialized

   (d) Based on the data required for the first 30 minute dynamic timestep, meteorological data for the relevant dates are read into memory and distributed between the workers. This is described in more detail in section 3.4

   (e) The output grid, a 4-dimensional distributed array with 3 spatial dimensions and a fourth indexing the chemical species being modeled, is established

3. A skeleton emissions framework is established. Currently, J-Zephyr checks for any of 6 different hard-coded tracer names, and applies 100 $\mathrm{kg\,s^{-1}}$ emissions at a specific surface grid cell. The 6 tracers currently represent inert emissions from London, Houston, Beijing, Boston, Seattle and Johannesburg. As with all other arrays, these emissions are distributed between the workers

4. The first meteorological fields are read in. This step is special, as two instantaneous samples need to be read for interpolation, whereas at other times only one is needed (the previous 'end' arrays can be moved to 'start' arrays). This is also the only point at which data read-in might not align with the meteorological data's sampling times

5. Using the hybrid eta pressure level system employed by GEOS-Chem, the surface pressure data is extended to produce a surface-following pressure level system

6. Quantities such as grid box air mass, air density and geometric depth are estimated using hydrostatic approximations and ideal gas laws

7. The master loop is initialized; this loop runs in six-hour timesteps, reflecting the temporal resolution of the meteorological data. The loop continues until the specified end date is reached
   (a) The inner (dynamic timestep) loop is initialized, utilizing the timestep length specified in the input file
      i. At each inner timestep, the surface pressure is recalculated by interpolation between two instantaneous readings, with dependent pressure and air grids recalculated accordingly
      ii. Emissions are converted from $\mathrm{kg\,s^{-1}}$ of the given tracer to a mixing ratio (moles of X per mole of air) and added to the tracer mixing ratio array
      iii. Horizontal and vertical transport resulting from simple advection is calculated
      iv. Planetary boundary layer mixing is applied, using GEOS-Chem's simple `TURBDAY` scheme which forces uniform mixing ratios to be applied in each column within the PBL
      v. Timing data is updated to reflect the inner timestep completion
   (b) The updated mixing ratio array is written to a set of external files, appending for each 6-hour timestep completion
8. Any open files are closed, and J-Zephyr exits

3.2 Transport code details

The transport code is an almost direct port of the `TPCORE FVDAS` model found in GEOS-Chem, more information about which can be found on the Harvard GEOS-Chem website. This is a state-of-the-art transport code, and as such the implementation was non-trivial; consequently, implementation of the bare bones of this code was completed only shortly before the project deadline, with J-Zephyr totalling approximately 5,600 lines (including comments and whitespace) at last count. The pertinent details of the code are that it operates in all 3 directions on the atmospheric grid; this is a particular parallelization challenge due to the fact that a rectangular grid with conventional base and ceiling boundary conditions has to also map grid cell 1 to $n + 1$ by latitude, while somehow dealing with singularities at both poles (in this case through use of a polar cap).

3.3 Domain decomposition

At the lowest level, a CTM relies on a large number of 'feeder' arrays to generate 'intermediate' and 'output' arrays. In the case of J-Zephyr, the intermediate arrays are a variety of 2-D and 3-D meteorological data sets, either averaged over 3 or 6 hour intervals or sampled once every 6 hours. The details of this sampling are not important from the perspective of this class, but choosing how best to handle the domain decomposition with these arrays in mind turned out to be critical.

Early in the project, a decision was made (following the lead of MPI implementations in other CTMs such as TOMCAT/SLIMCAT) to decompose the domain by both latitude and longitude. To achieve this, Julia's in-built distributed array generation functions were applied to an 'example' 2-D grid when initializing the model variables; this set of array splits was then applied to every subsequent data set loaded into the memory. Arrays with more than 2 dimensions were still only split by latitude and longitude; this preserved full atmospheric columns and the full set of chemical species, which is ideal from the perspective of parallelizing chemistry (a future objective) but the worst possible approach for transport.

This is best exemplified by the core data block of any CTM – the array of average cell mixing ratios. This array is $\lambda \times \phi \times z \times X$ in size, where λ is the longitude, ϕ the latitude, and z the altitude index for a grid box, with X the species in question. Many chemical processes will operate purely within a single grid cell but involve knowledge of many different species. For example, say that species X_1 and X_2 can react to form species X_3 via the reaction $\mathrm{X_1 + X_2 \longrightarrow 2\,X_3}$. We could safely parallelize over the 1st through 3rd dimensions of the mixing ratio array. This is complicated somewhat by the fact that many chemical and dynamical processes such as precipitation and photolysis, both of which are critical to corretly modeling pollution formation and removal, are columnar in nature; however, as long as the 3rd and 4th dimensions are not distributed, this is not a problem. This has resulted, historically, in a trend towards distributing data by latitude and longitude, an approach that was followed here.

However, chemical transport is ignorant of chemical species interactions – it is determined only by application of meteorological information to the last calculated set of species mixing ratios. It is also much faster in horizontal (λ and ϕ) directions than verticalm such that low-order methods (requiring

minimal communication) often suffice for the dominant mechanisms underlying vertical transport. As such, there is no communication required when distributing over X alone from a transport perspective, or at least reduced communication if distributing over z rather than any horizontal dimension. This inherent conflict of interests exemplifies the difficulties of succesfully parallelizing a CTM.

Any approach to parallelization which does not maintain a full horizontal grid for each core suffers from two further difficulties. Not only must there be communication between adjacent workers, this adjacency is complicated by the fact that the Earth is not flat. As mentioned earlier, the 'West-most' worker is still required to communicate with the 'East-most' worker, given that $+180° = -180°$ in terms of longitude. Worse yet, the naïve approach to domain decomposition taken early in J-Zephyr did not consider the problem of polar singularities. As one moves further North, the grid cell area collapses to zero; accordingly, the Courant-Friedrichs-Lewy (CFL) number $\frac{V}{\Delta x}$ tends towards infinity under all conditions with non-zero wind velocities. To prevent this resulting in solution instability, the top and bottom two bands are averaged to form 'polar caps', and a different transport scheme (semi-Lagrangian flux form) is used for all zonal bands which contain grid cells with CFLs over 1 (corresponding to solver instability under conventional Eulerian schemes). While this is fine, if irritating, in the case of a non-parallelized scheme, the polar averaging necessitates multiple synchronization barriers where they would not otherwise be necessary. Worse yet, the averaging operation corresponds to a great deal of worker communication.

3.4 Meteorological data

The underlying meteorological data for the model is held in either 2-D (horizontal) or 3-D (full) grids which are themselves held in `FORTRAN` formatted records. Each record is relevant to a single day, and can be one of the three types shown in table 1.

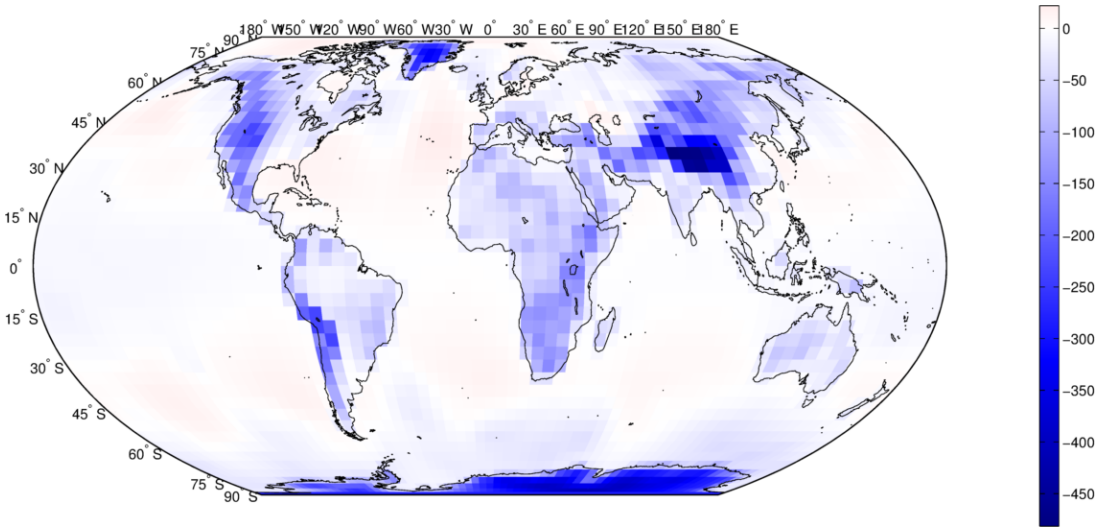
	A6	A3	I6
Sampling period (hours)	6	3	6
Averaged?	✓	✓	✗
1st read in time	21:00	00:00	21:00
1st sample time	00:00	01:30	21:00
Number of fields	21	33	5

Table 1: Meteorological data used in GEOS-Chem, derived from the NASA Global Earth Observation System (GEOS-5)

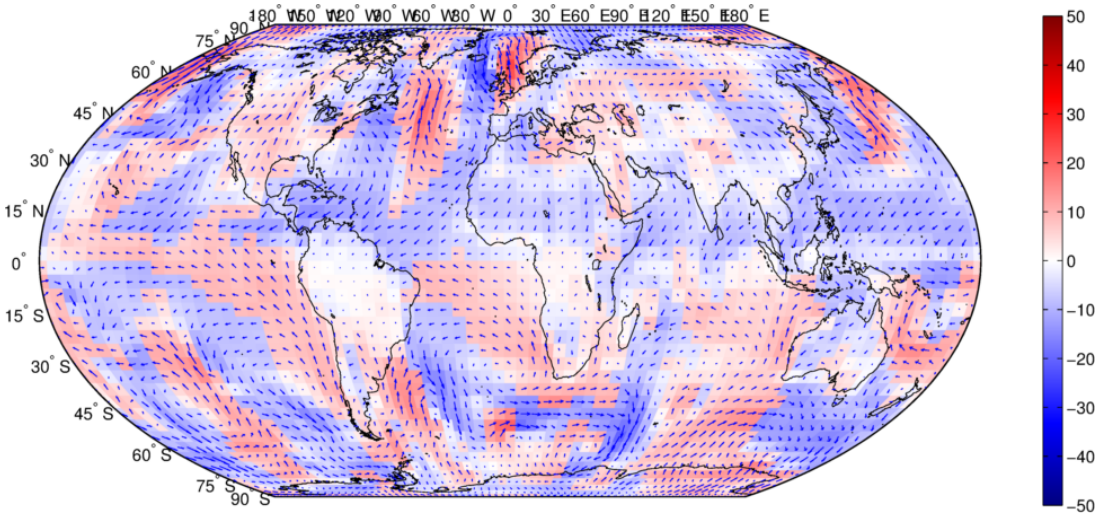
The instantaneously-sampled fields (I6) are interpolated between sampling points, where necessary, whereas the averaged fields are used (unmodified) throughout the relevant sampling period. As a result, each of the three different data sets must be read in at different times during the timestepping process. Figure 1 shows surface pressure (an I6 field) and surface wind velocity (A6 fields) from 2004-01-01, having been read in using the relevant J-Zephyr routines. Plotting was performed by reserializing the data into a simple ASCII file and then reading it into MATLAB to allow the proprietary mapping toolbox to be leveraged.

4 Results

The key function of J-Zephyr was to serve as a learning experience regarding the difficulties and unique roadblocks associated with parallel programming. However, the project did also produce results in the form of a partial transport code. If only to showcase the completion of the project, figure 2 shows estimated concentrations (in parts per billion by volume, equivalent to nmol pollutant per mol air) of inert pollutants with emissions rates of 100 $\mathrm{kg\,s^{-1}}$ from London, Houston, Beijing, Boston, Seattle and Johannesburg, beginning at the start of 2004-01-01 (GMT). The readings are taken after eight simulated hours of emissions, mixing and transport. As an example of the capabilities of such a model, notice the disparity between the behaviour of emissions in Johannesburg and in London. The Johannesburg emissions remain tightly contained, resulting in very high local surface mixing ratios. By contrast, the emissions from London are rapidly dispersed, albeit across the rest of the UK, resulting in lower peak mixing ratios. Comparison to figure 1 shows that this behaviour is the result of circulating, low-speed flow in South Africe, whereas around London there are high wind speeds and significant large-scale wind shear. The relative impacts depend on the species in question; some health impacts have been linked to



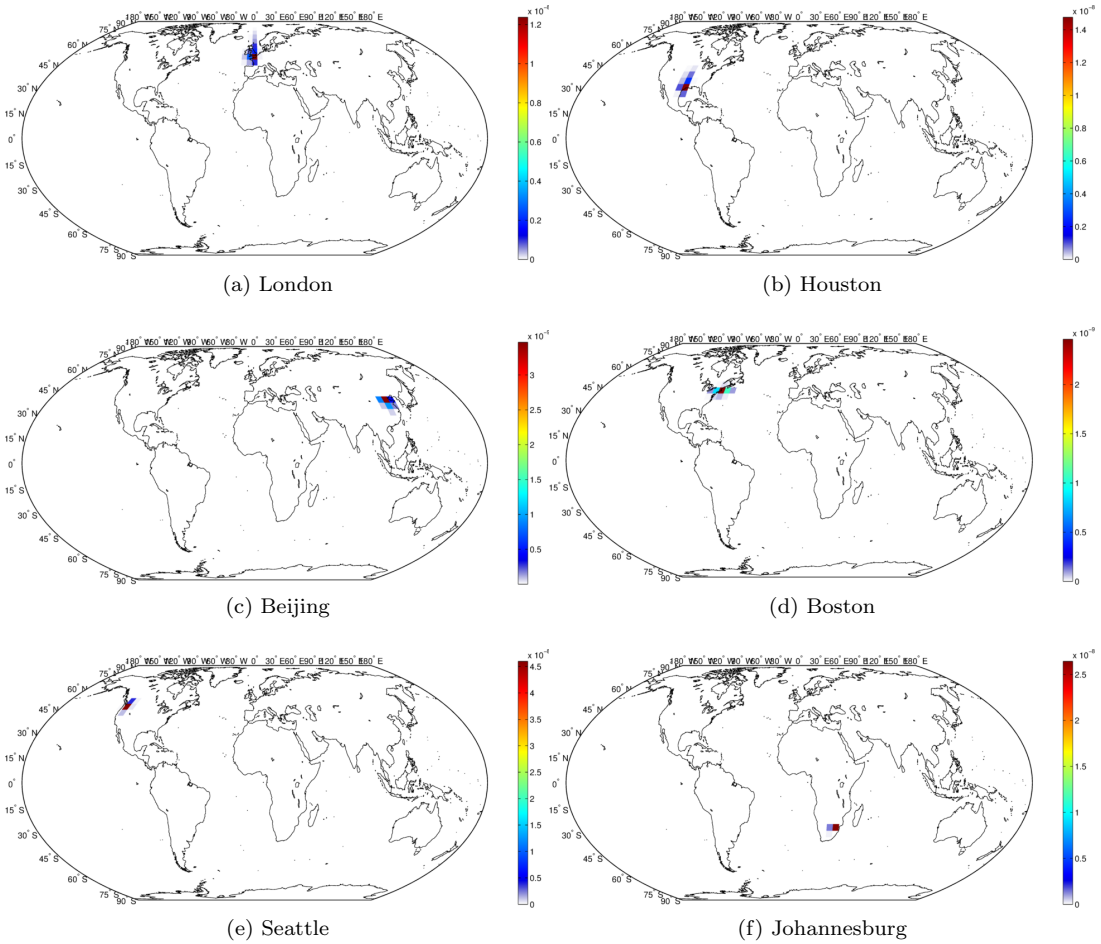
(a) Surface pressure anomaly (hPa)



(b) Wind velocity ($\mathrm{m\,s^{-1}}$)

Figure 1: Meteorological fields read in from GEOS-5. Surface pressure is shown relative to a nominal average value of 1013 hPa, while the wind velocity is shown in both absolute terms and as a vector. The sign of the wind field corresponds to whether the net movement is North (red, positive) or South (blue, negative). This plot highlights the presence of the Inter-Tropical Convergence Zone (ITCZ) at the equator, where the heating of the equator causes air to rise, drawing in air from above and below the equator; this is a central driver in atmospheric transport.

a threshold value (where very high mixing ratios are disproportionately more dangerous than moderate values), while others (such as particulate matter) may 'saturate' in terms of their health impacts (see reports from the WHO and EPA).



(a) London (b) Houston

(c) Beijing (d) Boston

(e) Seattle (f) Johannesburg

Figure 2: Surface mixing ratios (ppbv) after eight hours of emission and transport of an inert chemical from each of the listed cities.

Figure 3 shows the zonally-averaged results by latitude and altitude when considering all of the tracers together. Note that the tracers have different molecular masses – emission of 100 kg of a low molecular mass tracer will result in a higher overall mixing ratio than 100 kg of a high molecular mass tracer. Note that the model calculates transport up to an altitude of approximately 80 km, but only the lowest levels are shown here.

These figures, taken together, show that J-Zephyr is succesfully performing transport and PBL mixing in both horizontal and vertical directions. Independent checks built into the code show that overall mass is concerved by the PBL mixing operation, and that transport introduces an error of the order of 0.1 to 1% in each timestep. This error is likely the result of machine precision (especially since the mass total was naïvely calculated by sequential summation of grid cell masses). The only physical behaviour governing the predicted total tracer mass is the addition of $E = 100\ \mathrm{kg\,s^{-1}}$ of each tracer; comparing the calculated tracer mass after transport to the total expected as a result of the simple calculation $\Delta t \times E$, where Δt is the time elapsed (in this case $8 \times 60 \times 60 = 28800$ seconds), the error appears to be stable at the 0.1 to 1% level.

5 Conclusions

5.1 Deliverables

The following goals were laid out in the original proposal, and were all successfully completed:

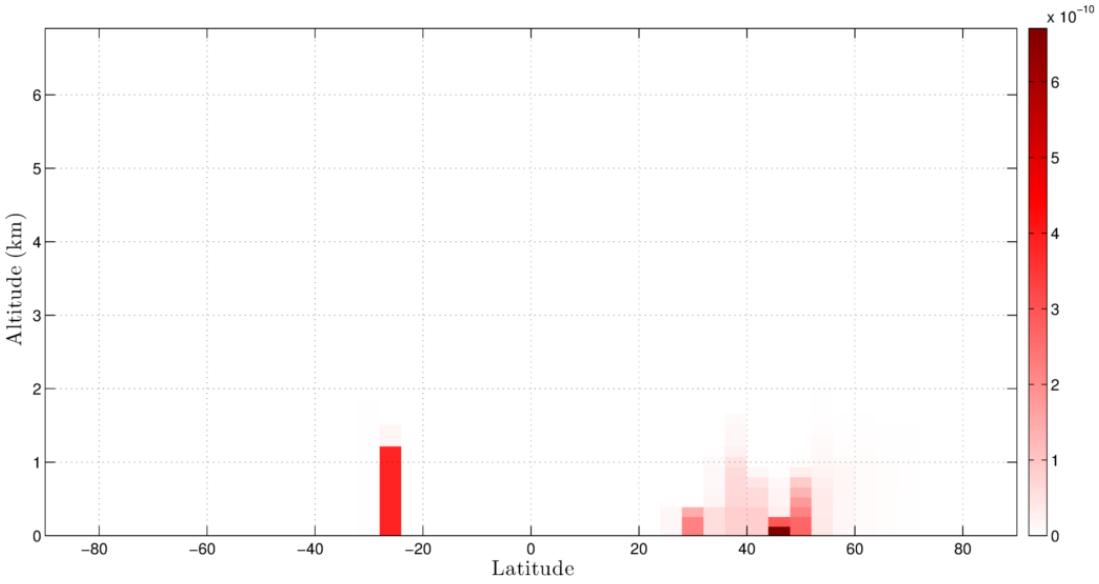


Figure 3: Zonal average results of all the tracers put together. Only the lowest set of pressure levels are shown, for the sake of clarity.

- Reads GEOS-5 meteorological data at a 4° latitudinal and 5° longitudinal resolution over 72 hybrid-eta pressure layers
- Distributes the necessary fields between workers (NB: currently only on one machine)
- Full atmosphere discretized on the same horizontal grid (discretized by longitude and latitude) and vertical grid (hybrid eta) as the source data
- Calculates advective transport using a 30 minute discretization for an arbitrary number of fields over at least 1 simulated year

In addition to these core goals, basic emissions were also implemented, and the model's flexibility (implemented through a user input file) exceeds the original specifications. Furthermore, the stretch goal of implementing PBL mixing was also achieved.

5.2 Lessons

This project was an excellent learning experience from the perspective of familiarization not only with parallel programming techniques but also with the Julia language. Firstly, Julia's parallelization macros make the process of parallelizing small sections of code fast, but causes difficulties when producing larger models. In particular, fine-grained control of the sort which is necessary to ensure high speed and to keep a tight control on memory usage on various machines is difficult.

More generally, the demands made by the model in terms of both communication and computation exemplified the challenges of parallel computing. The most significant lesson learned from this project is that careful domain decomposition could be critical to the success or failure of a project. In the case of this project, even assuming that spatial distribution was preferable to parallelization across species, the use of even a simple scheme which could preserve polar caps on a minimal number of workers would have significantly reduced the communication overhead.

5.3 Future work

I intend to continue working on J-Zephyr as a side project. Although there are essentially an unlimited set of possible new features to be implemented, I consider the following three items to be the highest priorities:

- Machine file reading and multi-machine setup. There are no significant obstacles to this implementation, beyond simply the time investment involved in learning Julia's cluster management

- Deep convective mixing. This form of mixing is responsible for large-scale movement of species from the surface into the upper atmosphere due to equatorial convection; without it, vertical mixing (and subsequent large-scale dispersion) is severely underestimated
- Domain decomposition overhaul. I intend to return to the original domain decomposition and attempt to restructure the code in such a way that there is less wastage, with a strong focus on reducing worker communication

5.4 Acknowledgements

I would like to thank Jiahao Chen and Jeff Bezanson for their assistance as I worked my way through this projec, in particular for their help in diagnosing the problems with memory management and variable scope which so delayed the project.

A Original Proposal

A.1 Introduction and Objective

The global chemistry-transport model (CTM) is a specialized tool for analysis of chemical perturbations to the Earth's atmosphere. The basic unit of a CTM is the grid cell, typically defined in terms of a range of latitudes, longitudes and pressures. Each grid cell is considered to contain almost exclusively air in the form of 21% oxygen and 79% nitrogen, with everything else considered to be a 'trace species' and described in terms of its ratio to the amount of air in the grid cell. At its simplest, pre-computed horizontal wind fields are applied to calculate the flux of air into and out of a grid cell with respect to its 'in-layer' neighbors, while vertical flux is calculated based on the much weaker vertical winds. Molecular diffusion can generally be ignored below around 100 km. These meteorological fields are supplied based on historical reanalyses of observations by general circulation models (GCMs), which directly address large-scale atmospheric dynamics but tend to ignore or simplify atmospheric chemistry to a much greater extent than CTMs.

To phrase this mathematically, we specify the continuity equation for a point in space. C is the 'mixing ratio' of the species of interest to air, n_a is the number density (molecules per m^3) of air, $\vec{U}$ is the wind velocity field (ms^{-1}), P is the chemical production of the species in molecules cm^{-3} s^{-1} and L is the loss rate in the same units:

$$\frac{\partial C n_a}{\partial t} = -C\nabla \cdot \left(C n_a \vec{U}\right) + P - L$$

Typically, CTMs use the concept of 'operator splitting' to reduce this equation into two steps; a transport step and a chemistry step. The transport step (equation 1) solves for the movement of air between grid boxes, ignoring chemical processes. The chemistry step (equation 2) estimates the chemical production and loss within that grid cell, ignoring bulk transport.

$$\frac{\partial n}{\partial t} = \left[\frac{\partial n}{\partial t}\right]_{\text{transport}} + \left[\frac{\partial n}{\partial t}\right]_{\text{chemistry}}$$

$$\left[\frac{\partial n}{\partial t}\right]_{\text{transport}} = -\nabla \cdot \left(n\vec{U}\right) \tag{1}$$

$$\left[\frac{\partial n}{\partial t}\right]_{\text{chemistry}} = P - L \tag{2}$$

These operations can be applied on a discretized grid by defining separated transport and chemical operators $\vec{T}$ and $\vec{C}$ such that

$$n\left(t_0 + \Delta t\right) = \vec{C} \cdot \vec{T}\left(n\left(t_0\right)\right)$$

where $\vec{C}$ and $\vec{T}$ independently solve the chemistry and transport equations over Δt. For this project, I will attempt to recreate the transport module of a global CTM, forming the basis of a new, Jula-based CTM – J-Zephyr.

Note: this introduction drew heavily on the webpage `http://acmg.seas.harvard.edu/education/jacob_lectures_ctms_chap1.pdf`, which goes into significantly more detail on the subsubject.

A.2 Approach

A.2.1 Basis

The MIT Laboratory for Aviation and the Environment (LAE) makes extensive use of the community-developed, open-source global CTM GEOS-Chem (originally developed and still maintained by Harvard University's School of Environmental and Atmospheric Sciences, SEAS). This model, written in FORTRAN, is capable of accepting a variety of meteorological fields, including the NASA Global Modeling and Assimilation Office (GMAO) Global Earth Observation System (GEOS) reanalyses. While GEOS-Chem is written in FORTRAN for speed, it has only limited parallelization, restricted to multiple-core single-machine runs using OpenMP.

I intend to use the code approach taken in GEOS-Chem and adapt it to Julia. Since Julia and FORTRAN are fundamentally different in terms of coding style, I will be basing my code structure on GEOS-Chem but starting from scratch rather than trying to copy and adapt the existing code.

A.2.2 Challenges

GEOS-Chem uses OpenMP, with a single thread farming tasks out to n workers (including itself) on a single machine. The primary challenge of implementing J-Zephyr will not be the structure of a CTM itself, which has been written many times before. Instead, J-Zephyr is to be arbitrarily parallel. A single head processor, hereafter known as 'root', will be used only for data acquisition and process coordination. This root processor will be in control of n workers distributed irregularly over k machines. The overall domain will be divided equally between these workers using distributed arrays to hold the relevant data. Doing this in a Lagrangian model would be significantly easier – solving the transport of m non-interacting parcels of air can be done without worker communication. However, for an Eulerian model, each worker must communicate with its 'neighbors' (based on each worker handling a set of zonal bands circumscribing the Earth). Achieving this without excessive communication through the head node will be difficult.

A.2.3 Workflow

The following are considered to be the deliverables of the project:

1. Data read function. Function to read all 2-D and 3-D fields from a single meteorological data file. Must accept 5 types:
    - `A1` – Averages, 1-hour period
    - `A3` – Averages, 3-hour period
    - `A6` – Averages, 6-hour period
    - `I3` – Instantaneous samples, 3-hour period
    - `I6` – Instantaneous samples, 6-hour period
2. Base distributive program. Reads data on the head node, distributing to each worker, running a function remotely on each worker and returning results to the head node
3. Base communicative program. After the distribution step from before, each worker must be able to request data from its neighbors and perform an operation which combines this data without communication through the head node
4. Forward model. Wrap these codes in a program for the head node which marches forward in simulation time, reading meteorological data at necessary intervals and using the distributed workers to determine a simple quantity. The example used will be vertical pressure distribution in the hybrid-eta system (no inter-node dependence but requires knowledge of the interpolated surface pressure at each time step)
5. Memory footprint evaluation. Add distributed arrays for all necessary quantities, including an arbitrarily-sized tracer mixing ratio array (containing j tracers) and all meteorological fields, to estimate the memory footprint of the program
6. Simulation interface. Add an interface for a simple input file to allow the initial distributions of the tracers to be set, along with information such as the location of the meteorological data, grid resolution, and transport timestep length. The program must return an error if the CFL condition is expected to be violated (based on a hard-coded prior estimate of the peak mean atmospheric windspeeds for a given grid resolution)
7. Transport model. Combine the capabilities above to model transport without PBL mixing or convective transport for j tracers between January 1st 2004 and January 1st 2005, comparing to results simulated in GEOS-Chem using an inert tracer also without PBL mixing or convective transport

A.2.4 Goals

The following capabilities are **primary objectives** for J-Zephyr, considered mandatory for project success:

- Reads GEOS-5 meteorological data at a 4° latitudinal and 5° longitudinal resolution over 72 hybrid-eta pressure layers

- Distributes the necessary fields between workers
- Full atmosphere discretized on the same horizontal grid (discretized by longitude and latitude) and vertical grid (hybrid eta) as the source data
- Calculates advective transport using a 30 minute discretization for an arbitrary number of fields over at least 1 simulated year

There are also a set of optional **secondary objectives**, considered as 'stretch goals' but unlikely to be completed:

- Simplified planetary boundary layer (PBL) mixing
- Convective mixing using cloud and humidity parameterizations
- Chemistry operator stub working on a different timestep and with a prescribed decay time for each tracer
- Dry deposition of chemical species based on known properties at ground level