FINAL REPORT

Uncertainty analysis and improvement of STILT-ASP for determining O$_3$ formation from biomass burning

TCEQ Contract No. 582-19-90498
Work Order No. 582-19-92805-03
Revision 2.0

Prepared by:
Benjamin Brown-Steiner, Archana Dayalu, and Matthew Alvarado
Atmospheric and Environmental Research, Inc. (AER)
131 Hartwell Ave.
Lexington, MA 02421-3126

Correspondence to: malvarad@aer.com

Prepared for:
Erik Gribbin
Texas Commission on Environmental Quality
Air Quality Division
Building E, Room 342S
Austin, TX 78711-3087

June 30, 2019
### Document Change Record

<table>
<thead>
<tr>
<th>Revision</th>
<th>Revision Date</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>30 May 2019</td>
<td>Internal Version for Review</td>
</tr>
<tr>
<td>1.0</td>
<td>1 June 2019</td>
<td>Draft Submitted to the TCEQ</td>
</tr>
<tr>
<td>2.0</td>
<td>30 June 2019</td>
<td>Final Report Submitted to TCEQ</td>
</tr>
</tbody>
</table>
List of Figures

Figure 1: STILT-ASP CO results before (left) and after (right) the PlumeRise algorithm was added. .......... 13
Figure 2: STILT-ASP O3 results before (left) and after (right) the PlumeRise algorithm was added. .......... 13
Figure 3: Comparison of KRR with OLS linear regression. ............................................................. 15
Figure 4: Representation of STILT-ASP sensitivity parameter space (see Table 2) with ΔO3 mixing ratios. .... 17
Figure 5: O3 emulation from OLS MLS (left) and KRR (right) over limited parameter space ...................... 19
Figure 6: OLS MLR (left) and KRR emulation (right) over parameter space. Colors are O3 ..................... 20
Figure 7: As in Figure 3 but for ΔO3 instead of O3. The green point inside is used as a validation point. ....... 21
Figure 8: As in Figure 4 but for ΔO3 instead of O3. The green point inside is used as a validation point. ....... 21
Figure 9: STILT-ASP simulated parameter space ................................................................................... 24
Figure 10: STILT-ASP ΔO3 for full simulated parameter space ............................................................... 25
Figure 11: KRR (left) and MLR (right) emulations for expanded parameter space .................................... 25
Figure 12: 3D reprojections of expanded parameter space from Figure 3 .............................................. 26
Figure 13: 2D “top-down” reprojection of STILT-ASP emulation .......................................................... 27
Figure 14: 2D “from the side” reprojections of STILT-ASP for KRR emulation ....................................... 28
Figure 15: 2D “from the side” reprojections of STILT-ASP for MLR emulation ...................................... 29
Figure 16: ΔO3 for each of 500 particles at each 15-minute timestep of base case STILT-XPLEX and NONXPLEX
48h forward model simulations (96000 points). (a) Comparison of NONXPLEX (black) and XPLEX (red)
on each 15-minute timestep of base case STILT-XPLEX and NONXPLEX forward model simulations (96000 points). (a) Comparison of NONXPLEX (black) and XPLEX (red)
along 48h of the simulation; XPLEX point size reduced for visual clarity (b) comparison of XPLEX vs. NONXPLEX with 1:1 line for comparison, (c) NONXPLEX-XPLEX residuals, and (d) histogram of residuals. ... 35
along 48h of the simulation; XPLEX point size reduced for visual clarity (b) comparison of XPLEX vs. NONXPLEX with 1:1 line for comparison, (c) NONXPLEX-XPLEX residuals, and (d) histogram of residuals. ... 35
Figure 17: ΔCO for each of 500 particles at each 15-minute timestep of base case STILT-XPLEX and NONXPLEX
along 48h of the simulation; XPLEX point size reduced for visual clarity (b) comparison of XPLEX vs. NONXPLEX with 1:1 line for comparison, (c) NONXPLEX-XPLEX residuals, and (d) histogram of residuals. ... 36
along 48h of the simulation; XPLEX point size reduced for visual clarity (b) comparison of XPLEX vs. NONXPLEX with 1:1 line for comparison, (c) NONXPLEX-XPLEX residuals, and (d) histogram of residuals. ... 36
Figure 18: ΔNO for each of 500 particles at each 15-minute timestep of base case STILT-XPLEX and NONXPLEX
along 48h of the simulation; XPLEX point size reduced for visual clarity (b) comparison of XPLEX vs. NONXPLEX with 1:1 line for comparison, (c) NONXPLEX-XPLEX residuals, and (d) histogram of residuals. ... 37
along 48h of the simulation; XPLEX point size reduced for visual clarity (b) comparison of XPLEX vs. NONXPLEX with 1:1 line for comparison, (c) NONXPLEX-XPLEX residuals, and (d) histogram of residuals. ... 37
Figure 19: ΔNO2 for each of 500 particles at each 15-minute timestep of base case STILT-XPLEX and NONXPLEX
along 48h of the simulation; XPLEX point size reduced for visual clarity (b) comparison of XPLEX vs. NONXPLEX with 1:1 line for comparison, (c) NONXPLEX-XPLEX residuals, and (d) histogram of residuals. ... 38
along 48h of the simulation; XPLEX point size reduced for visual clarity (b) comparison of XPLEX vs. NONXPLEX with 1:1 line for comparison, (c) NONXPLEX-XPLEX residuals, and (d) histogram of residuals. ... 38
Figure 20: Δ PAN for each of 500 particles at each 15-minute timestep of base case STILT-XPLEX and NONXPLEX
along 48h of the simulation; XPLEX point size reduced for visual clarity (b) comparison of XPLEX vs. NONXPLEX with 1:1 line for comparison, (c) NONXPLEX-XPLEX residuals, and (d) histogram of residuals. ... 39
along 48h of the simulation; XPLEX point size reduced for visual clarity (b) comparison of XPLEX vs. NONXPLEX with 1:1 line for comparison, (c) NONXPLEX-XPLEX residuals, and (d) histogram of residuals. ... 39
List of Tables

TABLE 1. Projected Schedule for TCEQ Work Order No. 582-19-92805-03 ................................................................. 10
TABLE 2: Summary of STILT-ASP Parameter Sensitivity Simulations ................................................................. 16
TABLE 3: Select Gas Species Results from STILT-ASP Sensitivity Simulations .................................................. 18
TABLE 4: Net O₃ Difference for Different Combinations of Sensitivity Simulations ........................................ 18
TABLE 5: Parameter Values and ΔO₃ (ppb) for All 23 STILT-ASP Simulations ............................................. 23
TABLE 6: Summary of Results from Perturbed and Non-Perturbed STILT-ASP_XPLEX Simulations .......... 34
List of Acronyms

AER – Atmospheric and Environmental Research
ASP – Aerosol Simulation Program model
BBOP – Biomass Burning Observation Project
CAMx – Comprehensive Air Quality Model with Extensions
CTM – chemical tracer model
CO – carbon monoxide
DDM – Direct Decoupled Method
FINN – Fire INventory from NCAR
GEOS-Chem – Goddard Earth Observing System Chemistry Model
HGB – Houston-Galveston-Brazoria
KRR – kernel ridge regression
MEGAN – Model of Emissions of Gases and Aerosols from Nature
MLR – multiple linear regression
NARR – North American Regional Reanalysis
NOx – nitrogen oxides (NO + NO2)
O3 – ozone
OLS – ordinary least squares
PAN – peroxyacetyl nitrate
PBL – planetary boundary layer
PM2.5 – fine particulate matter
RMSE – root means square error
R2 – “R-Squared” Correlation
RR – ridge regression
STILT – Stochastic Time-Inverted Lagrangian Transport model
STILT-ASP – joint STILT and ASP model
VOC – Volatile Organic Compounds
WRF – Weather Research and Forecasting
WRF-Chem – Weather Research and Forecasting Chemistry model
XPLEX – Complex Step Approach
ΔO3 – change in O3 between fire and no-fire STILT-ASP simulations
Executive Summary

The purpose of this project was to perform a global uncertainty analysis of the current STILT-ASP model via an emulation approach and a complex step approach, as well as to develop parameter updates for plume rise and mixing parameterizations. All our tasks used the University of Texas El Paso fire event on June 21, 2015 as the case-study.

In Task 2, an improved plume height algorithm for fires, used in the WRF-Chem model, was added to STILT-ASP. However, this new plume rise algorithm had little to no impact on the STILT-ASP results for our example case, as most of the plume injection heights were calculated to be below the 2000 m minimum mixing height used in STILT-ASP. Tasks 3 and 4 built upon the updated version of STILT-ASP developed in Task 2.

In Task 3, kernel ridge regression (KRR) was used to create a statistical emulator of STILT-ASP and a global uncertainty study was conducted to identify the key input parameters that need examination to further improve model performance. This global uncertainty study used both KRR and the multiple linear regression (MLR) technique to examine four parameters within STILT-ASP with expected O₃ controls: a fire size/plume rise parameter, the fire NOₓ emissions, the minimum planetary boundary layer (mixing) height parameter, and the ozone dry deposition parameter. While the fire size parameter demonstrated a slight difference in the vertical distribution of biomass burning plumes, there was a negligible influence on modeled O₃ concentrations and thus this variable was omitted from the emulation approaches. The other three parameters demonstrated a moderate impact on O₃ mixing ratios (~ -0.1 – 1.0 ppb when the parameters were modified by a factor of 2), with the fire NOₓ emissions having the largest impact. While the KRR emulation technique is known to be better at representing non-linear systems than a more standard multiple linear regression approach, the parameter space explored in this study had a largely linear impact on O₃ mixing ratios, and so both the KRR and MLR emulation techniques demonstrated similar behavior.

In Task 4, a complex step sensitivity approach was applied to the STILT-ASP model and a local sensitivity study was conducted to determine the sensitivity of modeled concentrations of several species to small changes in the magnitudes of fire, biogenic, and anthropogenic emissions. The concentrations from the complex step STILT-ASP version do not exactly match the original source code. While the small errors (less than 0.01%) in the gas concentrations is likely due to the change from single precision to double precision for some variables, the larger (~2%) differences in PM₂.₅ may be due to an unresolved bug in the code and require further investigation (Section 4.3).

However, the sensitivities of O₃, PM₂.₅, and other species relative to fire, anthropogenic, and biogenic emissions calculated using the complex-step approach are consistent with our expectations. For example, O₃ is most sensitive to perturbations in biogenic and anthropogenic emissions. Relative to biogenic emissions, O₃ sensitivity to equivalent perturbations of anthropogenic and fire emissions is lower by 33% and 80%, respectively. On the other hand, PM₂.₅ is most sensitive to fire emissions (Section 4.3).

We recommend that future work developing STILT-ASP should focus on other uncertain parameters such as increasing the resolution of the emissions and meteorology, improvements in the FINN fire emissions, and improvements in the chemistry in ASP. Future work should explore the emulator’s capability to estimate STILT-ASP output for different sites and at different times of year. In addition, the complex-step version of STILT-ASP should be used to investigate the
sensitivity of the predicted fire $O_3$ contribution to other uncertain variables that are usually difficult to perturb in 3D photochemical simulations (e.g., meteorological variables).
1. Introduction

1.1 Project Objectives

The purpose of this project was to perform a global uncertainty analysis of the current STILT-ASP model (using a statistical emulator approach) to identify the key input parameters that need to be studied to further improve model performance. A version of the STILT-ASP model that uses the complex step approach to estimate the sensitivity of O₃ formation to fire emissions and other inputs was developed. In addition, an improved plume height and mixing parameterization for fires was added to STILT-ASP. The deliverables include the code for the improved STILT-ASP model, the STILT-ASP emulator, and a report summarizing the results of the uncertainty analysis and the impact of the improvements made to STILT-ASP in this project. All our tasks used the University of Texas El Paso fire event on June 21, 2015 as the case-study.

In Task 2 of this project, AER added a parameterization for the plume rise of biomass burning emissions to STILT-ASP following the approach used in the WRF-Chem fire emissions preprocessor with adjustments made based on recent literature studies. The impact of the new plume rise parameterization on the STILT-ASP predictions of the fire impacts on O₃ and PM₂.₅ were characterized.

In Task 3, AER used the KRR approach to develop an emulator of the STILT-ASP model and used the emulator to perform a global study of the impact of the uncertainty in key model parameters (e.g., mixing height, plume injection height, NOₓ:VOC ratios in the biomass burning plumes, deposition velocity) on the estimates of the fire impacts on O₃ and PM₂.₅ at urban areas in Texas. To develop the training data set for the emulator, AER determined the uncertainty ranges in these parameters for conditions in Texas. In order to perform a full global sensitivity analysis with the emulator, the training data set went beyond the range of reported uncertainty. Based on this determination, AER performed a series of STILT-ASP simulations spanning the uncertainties within the parameters and examined the STILT-ASP emulator capabilities and impacts on O₃.

In Task 4, AER implemented the complex step approach in STILT-ASP following the methods used by Constantin and Barrett (2014) in GEOS-Chem. This new code was used to explore the sensitivity of O₃ and PM₂.₅ to emissions from biomass burning, anthropogenic, and biogenic sources for cases previously studied with STILT-ASP by Brown-Steiner et al. (2018). A technical memo and user’s guide for the complex step approach was produced.

The schedule of deliverables for this project is given in Table 1.
Table 1. Projected Schedule for TCEQ Work Order No. 582-19-92805-03

<table>
<thead>
<tr>
<th>Milestones</th>
<th>Planned Date</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Task 1 – Work Plan</strong></td>
<td></td>
</tr>
<tr>
<td>1.1: TCEQ-approved Work Plan</td>
<td>February 11, 2019</td>
</tr>
<tr>
<td>1.2: TCEQ-approved QAPP</td>
<td>February 11, 2019</td>
</tr>
<tr>
<td><strong>Task 2 - Update plume rise and mixing parameterizations</strong></td>
<td></td>
</tr>
<tr>
<td>2.1: Technical memo summarizing the new plume rise algorithm</td>
<td>March 31, 2019</td>
</tr>
<tr>
<td><strong>Task 3 – Kernel Ridge Regression Emulator and Global Uncertainty Study</strong></td>
<td></td>
</tr>
<tr>
<td>3.1: Technical memo describing the KRR methodology and its applicability to the STILT-ASP model</td>
<td>April 30, 2019</td>
</tr>
<tr>
<td><strong>Task 4 – Complex Step Approach Emission Uncertainty Study</strong></td>
<td></td>
</tr>
<tr>
<td>4.1: Technical memo describing the complex step approach and its applicability to the STILT-ASP model</td>
<td>April 30, 2019</td>
</tr>
<tr>
<td>4.2: User’s guide for the complex step approach describing the complex step code, its hardware requirements, and how to use the updated code to calculate first-order sensitivity parameters.</td>
<td>April 30, 2019</td>
</tr>
<tr>
<td><strong>Task 5 – Draft and Final Reports</strong></td>
<td></td>
</tr>
<tr>
<td>5.1: Draft Report</td>
<td>June 1, 2019</td>
</tr>
<tr>
<td>5.2: Final Report</td>
<td>June 30, 2019</td>
</tr>
<tr>
<td>5.3: STILT-ASP Revisions</td>
<td>June 30, 2019</td>
</tr>
</tbody>
</table>

1.2 Background

Over the past few years, the U.S. Environmental Protection Agency (EPA) has set increasingly stringent National Ambient Air Quality Standards (NAAQS) for ozone (O$_3$) and fine particulate matter (PM$_{2.5}$). With these lower standards, it has become increasingly likely that discrete events such as wildfires will cause an area to violate the O$_3$ and/or PM$_{2.5}$ NAAQS.

The STILT-ASP model is a simplified photochemical model that can be used to determine the impact of biomass burning on O$_3$ and PM$_{2.5}$ at specific monitors for exceptional event demonstrations. However, the current STILT-ASP model includes many uncertain input parameters, and further improvement of the model requires an assessment of the relative impacts of these uncertain parameters on the modeled predictions of O$_3$ and PM$_{2.5}$ from fires.

This project applied two methods to the STILT-ASP model to perform parametric uncertainty assessment. First, AER used the STILT-ASP model to develop a training dataset for a Kernel Ridge Regression (KRR) machine-learning simulator. KRR has been widely and successfully used in many regression and function approximation tasks (Rivera et al., 2015; Camps-Valls et al., 2009; Mallet et al., 2009). The resulting emulator provides a new method to estimate the sensitivity of modeled O$_3$ concentrations to uncertainties in STILT-ASP inputs.

Second, AER also used the complex step approach (Squire and Trapp, 1998; Martins et al., 2003; Constantin and Barrett, 2014) to develop a version of the STILT-ASP model that can directly estimate the sensitivity of O$_3$ formation to fire emissions and other inputs. The advantage of the

10
complex step approach for this work is that it is easy to maintain as new scientific algorithms are added to the model.

1.3 Report Outline

This Final Report highlights major activities and key findings, provides pertinent analysis, describes encountered problems and associated corrective actions, and details relevant statistics including data, parameter, or model completeness, accuracy and precision. It satisfies Deliverable 5.2 of the Work Plan for Work Order No. 582-19-92805-03:

**Deliverable 5.2:** Final Report  
**Deliverable 5.2 Due Date:** June 28, 2019

The major results of Tasks 2, 3 and 4 are summarized in Sections 2, 3, and 4, respectively. Section 5 discusses the quality assurance findings for this project following the procedures from the project Quality Assurance Project Plan (QAPP). Section 6 summarizes our conclusions and Section 7 makes recommendations for future work based on the results of this project.
2 Updates to the plume rise and mixing parameterizations

2.1 Introduction

The purpose of this task was to add an improved plume height and mixing parameterization for fires to STILT-ASP. A stand-alone biomass burning plume rise algorithm (Frietas et al., 2007) was extracted from the WRF-Chem model and adapted for use in AER’s STILT-ASP model. We have edited the source code of STILT-ASP and the fire emissions preprocessor to successfully incorporate the plume rise algorithm.

2.2 Implementation Overview

The overall purpose of incorporating the WRF-Chem version of the “plumerise” algorithm is to more accurately estimate the injection height of biomass burning plumes. The equations of the algorithm have been extensively documented in the literature (e.g., in Frietas et al., 2007; Frietas et al. 2010; Walter et al., 2016). In the current version of the STILT-ASP model, the plume top is set at 2000 meters and fire emissions are assumed to be well mixed below this height. The plumerise algorithm instead estimates an injection height window, providing a lower and upper bound for the plume based on assumptions about the minimum and maximum heat flux for four fire types (temperate forests, tropical forests, shrubs, and grasslands). The fire sizes are taken directly from the FINN fire inventory (Wiedinmyer et al., 2011) and are summed for all fires of the same type within the meteorological model grid box, as is done in WRF-Chem.

In our incorporation of the plume rise algorithm, we use our original method of assuming the emissions are well-mixed below 2000 meters when the upper bound of the plume as estimated by the plumerise algorithm is less than or equal to 2000 meters. When the upper bound of the plume is estimated by the algorithm as greater than 2000 meters, then the emissions of each species are distributed evenly in each NARR grid box within the injection height window. STILT-ASP parcels that pass through those grid boxes have these adjusted emissions due to fire added to the parcel concentrations.

Appendix A provides an overview of the changes made to the STILT-ASP source code to implement the new plumerise algorithm.

2.3 Example Results: University of Texas El Paso Simulation

We have run the new code for an initial example case of a five-day back trajectory from the University of Texas El Paso at 12 UTC on June 21, 2015.

Figure 1 Figure 2 show the comparisons between the CO and O$_3$ concentrations respectively, predicted using the old code and the new code. We examined the final output concentration files for each simulation and found no differences in concentrations between the two simulations. This is due to the fact that the plume top height calculated by the plumerise algorithm rarely exceeds 2000 meters, and for those fires the two model simulations are identical.
Without Plumerise Algorithm

With Plumerise Algorithm

Figure 1: STILT-ASP CO results before (left) and after (right) the plumerise algorithm was added.

Without Plumerise Algorithm

With Plumerise Algorithm

Figure 2. STILT-ASP O3 results before (left) and after (right) the plumerise algorithm was added.
3 KRR Emulation and Global Uncertainty Study Results

3.1 Background

The STILT-ASP model (e.g. Alvarado et al., 2016) is a simplified photochemical model that can be used to determine the impact of biomass burning on O$_3$ and PM$_{2.5}$ at specific monitors for exceptional event demonstrations. However, the current STILT-ASP model includes many uncertain input parameters. A full sensitivity study exploring the impact of model input parameters on O$_3$ and PM$_{2.5}$ would require a large ensemble of STILT-ASP simulations and would be costly and time consuming. One solution is to develop a statistical emulator that is trained on a representative ensemble of STILT-ASP sensitivity simulations, and then use the emulator to explore a wide range of model input parameters. These results will provide insight into the input parameters that can be used to inform future sensitivity simulations of the full STILT-ASP model.

In the remainder of this section we describe the KRR methodology and provide a simple example comparing the KRR approach with the more traditional ordinary least squares (OLS) linear regression approach.

3.2 Kernel Ridge Regression Emulation Technique

3.2.1 Kernel Ridge Regression (KRR)

Unlike ordinary least squares (OLS) linear regression, which determines a best-fit to any n-dimensional dataset by minimizing the squared difference between each point and the final fit, ridge regression (RR) is a regularization technique that incorporates an additional penalty proportional to the square of each regression coefficient (Hastie et al., 2017). This approach is particularly useful when performing regressions over a large number of variables (usually ten or more, which can be prone to overfitting). For each additional variable added to a standard linear regression, the magnitude of the regression coefficients can dramatically increase in magnitude. If this occurs the regression is overly sensitive to the model variables. Thus, while an overfit regression appears to be highly predictive over the data used to produce the regression, it has achieved this predictive capability by fitting to not just the underlying signal but also to the noise contained within the training data. Attempting to use an overfit regression to predict new data will produce (often highly) erroneous results. The addition of a penalty proportional to the square of each regression coefficient, as RR does, prevents extremely high regression coefficients and thus reduces the changes of an overfit regression.

Kernel ridge regression (KRR) applies one additional optimization called “the kernel trick.” This performs an additional data transformation by computing the inner products of all pairs of the training dataset. This transformation produces a linearly separable dataset from a dataset that is not linearly separable, and results in a highly skillful non-linear regression. In the following subsection we demonstrate this by comparing KRR with OLS regression (e.g. Rivera et al., 2015; Camps-Valls et al., 2009; Mallet et al., 2009).

---

1 [https://www.analyticsvidhya.com/blog/2016/01/complete-tutorial-ridge-lasso-regression-python/](https://www.analyticsvidhya.com/blog/2016/01/complete-tutorial-ridge-lasso-regression-python/)
3.2.2 Simple Comparison of KRR with OLS Linear Regression

Figure 3: Comparison of KRR with OLS linear regression.

Figure 3 compares KRR with OLS regression for a 1D artificial non-linear dataset. The thin red line represents a sinusoidal function to which white noise was applied for use in the regressions (orange and blue dots). These data points were separated into training (orange) and testing (blue) datasets with two-thirds of the data points being used to train the regression and one-third used for testing and evaluation. The purple line plots the OLS regression and is incapable of capturing the non-linear nature of the data. The green line overlaps with the original function (thin red line), and plots the KRR. As can be seen, KRR is highly capable of capturing the non-linear nature of the sinusoidal function. These capabilities, as shown below, extend to multidimensional datasets, although visualizing and conceptualizing the results for higher dimensional data can be challenging.
3.3 Preliminary STILT-ASP Simulations

Table 2: Summary of STILT-ASP parameter sensitivity simulations

<table>
<thead>
<tr>
<th>Simulation</th>
<th>NO\textsubscript{x} Scaling</th>
<th>PBL Scaling</th>
<th>O\textsubscript{3} Deposition Scaling</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0.5</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1.5</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0.5</td>
<td>1.5</td>
</tr>
<tr>
<td>5</td>
<td>1.5</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>1.5</td>
<td>0.5</td>
<td>1</td>
</tr>
<tr>
<td>7</td>
<td>1.5</td>
<td>1</td>
<td>1.5</td>
</tr>
<tr>
<td>8</td>
<td>1.5</td>
<td>0.5</td>
<td>1.5</td>
</tr>
<tr>
<td>9</td>
<td>1.25</td>
<td>0.75</td>
<td>1.25</td>
</tr>
</tbody>
</table>

We performed eight perturbation sensitivity simulations using three parameters within STILT-ASP: (1) increasing the NO and NO\textsubscript{2} emissions within biomass burning plumes by 50%; (2) reducing the mixing height by 50%; and (3) increasing the deposition velocity of O\textsubscript{3} by 50%. Table 2 contains the scaling values for all eight of the resulting parameter perturbations. We also increased the fire size and heat flux at first by a factor of two, and then by a factor of five, and while these perturbations did slightly change the heights at which the fire emissions were distributed, they did not result in changes in the final O\textsubscript{3} mixing ratios. We believe this lack of sensitivity is due to the fact that the plume rise height is mainly a function of the background atmospheric stability profile, rather than the fire parameters themselves. Thus, changing the fire size and heat flux does not significantly change the addition of emissions to the STILT parcels, and therefore does not change the final O\textsubscript{3} mixing ratios. Finally, we performed one final simulation in which we “split-the-difference” for each parameter (Simulation 9 in Table 2).

Figure 4 plots each of these eight simulations in a 3D representation of the parameter space for each of the three perturbed parameters with colors representing the O\textsubscript{3} value. This “box” represents the parameter perturbation simulations conducted with STILT-ASP and is included in the emulated parameter perturbation plots in the subsequent sections. We used a recent summertime STILT-ASP case in El Paso, Texas (latitude 31.768291°N and longitude -106.50126°E) from June 21, 2015. Table 3 contains the calculated mixing ratios for selected gas species for each simulation, while Table 4 summarizes the average effect from each of the scaled parameters. Values in multiple columns in each row of Table 3 represent a difference between two simulations in which only the parameters indicated in the row headings were different (thus there are different numbers of values for different combinations of scaled sensitivity parameters).

The highest O\textsubscript{3} – symbolized by the yellow dot in the top-left corner of Figure 4 – was obtained within simulation 6 (Table 3), with NO\textsubscript{x} scaled to 1.5, the PBL parameter scaled to 0.5, and the O\textsubscript{3} deposition left unscaled. The lowest O\textsubscript{3} – symbolized by the purple dot in the lower-right corner of Figure 4 – was obtained with the inverse set of parameters, with NO\textsubscript{x} and the PBL parameter left unscaled and the O\textsubscript{3} deposition scaled to 1.5. The difference between these high and low O\textsubscript{3} values was 82.21 – 81.34 = 0.86 ppb, which is 2-3x higher than the difference between each
simulation’s fire and no fire run (second column in Table 3). The average difference between fire and no fire simulations averaged over all eight of the perturbation simulations is 0.45 ppb.

Table 4 includes the perturbations between the various simulations for all possible combinations of scaling parameters and their averages, which are largely consistent from perturbation to perturbation. We can conclude from Table 4 that (1) increasing NOx emissions by 50% increases O3 by, on average, 0.27 ppb; (2) decreasing the PBL parameter by 50% increases O3 by, on average, 0.25 ppb; and (3) increasing the O3 deposition parameter by 50% decreases O3 by, on average, 0.34 ppb.

Figure 4: Representation of STILT-ASP sensitivity parameter space (see Table 2) with ΔO3 mixing ratios.
Table 3: Select gas species results from STILT-ASP sensitivity simulations.

<table>
<thead>
<tr>
<th>Simulation</th>
<th>O₃</th>
<th>ΔO₃</th>
<th>NO</th>
<th>NO₂</th>
<th>CO</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>81.68</td>
<td>0.236</td>
<td>0.210</td>
<td>0.676</td>
<td>146.59</td>
</tr>
<tr>
<td>2</td>
<td>81.90</td>
<td>0.399</td>
<td>0.225</td>
<td>0.724</td>
<td>149.58</td>
</tr>
<tr>
<td>3</td>
<td>81.34</td>
<td>0.235</td>
<td>0.211</td>
<td>0.676</td>
<td>146.59</td>
</tr>
<tr>
<td>4</td>
<td>81.57</td>
<td>0.398</td>
<td>0.226</td>
<td>0.724</td>
<td>149.59</td>
</tr>
<tr>
<td>5</td>
<td>81.92</td>
<td>0.478</td>
<td>0.213</td>
<td>0.687</td>
<td>146.59</td>
</tr>
<tr>
<td>6</td>
<td>82.21</td>
<td>0.705</td>
<td>0.230</td>
<td>0.739</td>
<td>149.59</td>
</tr>
<tr>
<td>7</td>
<td>81.59</td>
<td>0.477</td>
<td>0.214</td>
<td>0.686</td>
<td>146.59</td>
</tr>
<tr>
<td>8</td>
<td>81.87</td>
<td>0.703</td>
<td>0.231</td>
<td>0.739</td>
<td>149.59</td>
</tr>
<tr>
<td>9</td>
<td>81.80</td>
<td>0.388</td>
<td>0.219</td>
<td>0.705</td>
<td>147.63</td>
</tr>
</tbody>
</table>

Table 4: Net O₃ difference for different combinations of sensitivity simulations

<table>
<thead>
<tr>
<th></th>
<th>O₃ Differences between scaled and unscaled simulations</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOₓ scaled</td>
<td>0.24 0.31 0.24 0.31</td>
<td>0.27</td>
</tr>
<tr>
<td>PBL scaled</td>
<td>0.22 0.22 0.28 0.28</td>
<td>0.25</td>
</tr>
<tr>
<td>O₃ dep scaled</td>
<td>-0.34 -0.34 -0.33 -0.33</td>
<td>-0.34</td>
</tr>
<tr>
<td>NOₓ and PBL scaled</td>
<td>0.52 0.53</td>
<td>0.53</td>
</tr>
<tr>
<td>NOₓ and O₃ dep scaled</td>
<td>0.31 -0.09</td>
<td>0.11</td>
</tr>
<tr>
<td>PBL and O₃ dep scaled</td>
<td>-0.05 -0.12</td>
<td>-0.08</td>
</tr>
<tr>
<td>All parameters scaled</td>
<td>0.19</td>
<td>0.19</td>
</tr>
</tbody>
</table>
3.4 Preliminary KRR and MLR Emulation

KRR was applied to the scaling data in Table 2 and trained to predict the eight O₃ mixing ratios in Table 3 (with the exception of the ninth simulation which was used as an initial validation data point). We used default or recommended parameters for the regression.

3.4.1 Applying KRR Regression Emulator to Range of STILT-ASP Input Parameters

![Figure 5: O₃ emulation from OLS MLS (left) and KRR (right) over limited parameter space](image)

With the KRR emulator trained on the eight STILT-ASP data points in the three variables’ parameter space (Figure 4), we estimated O₃ mixing ratios for 1000 additional data points extending our scaling parameters out one additional “box length” in each direction (i.e. the NOₓ and O₃ deposition scaling was expanded from 1.0 – 1.5 to 0.5 – 2.0 while the PBL scaling was expanded from 0.5 – 1.0 to 0.0 – 1.5). These emulated results are plotted two ways for both OLS multiple linear regression and KRR.

First, in Figure 5, we display a 3D plot of the estimated O₃ mixing ratios (vertical axis) over the full spread of the O₃ deposition and PBL scaling parameters (black dots). In Figure 5, the original parameter “box” from Figure 4 is included with red triangles and is “squished” and “twisted” as the vertical axis in Figure 5 is the O₃ mixing ratios rather than the NO₂ scaling from Figure 4. Figure 5 demonstrates the clearly linear nature of the emulation using OLS MLR (left plot) in contrast to the distinctly non-linear nature of the KRR (right plot). Note that the vertical scales in both plots within Figure 5 are different, namely the OLS MLR estimates a smaller O₃ range than the KRR. The apparent vertical stacking of data points is due to the third parameter (NO₂ scaling) that is not included with these plots.

Second, in Figure 6, we plot the same data in the 3D parameter space identical to that in Figure 4 with the color scale representing the O₃ mixing ratios. The STILT-ASP parameter “box” form Figure 4 is included within Figure 6 (large spheres). Figure 6 shows a different view from Figure 5, but the same characteristics can be seen: the OLS MLR emulation results in a smaller O₃ spread and a linear O₃ mixing ratio gradient (from high values in the “upper left” and lowest values in the “lower right”), while the KRR shows a non-linear gradient over a much larger O₃ mixing ratio spread. In the KRR emulations, the highest O₃ mixing ratios tend to be centered within the
STILT-ASP parameter “box” and lower O₃ mixing ratios tend towards the outer edges and corners of the expanded emulated parameter box.

Figure 6: OLS MLR (left) and KRR emulation (right) over parameter space. Colors are O₃.

Finally, we performed a similar analysis but instead of emulating the total O₃ we emulated the ΔO₃ (i.e., the difference in O₃ between our simulations with fire emissions and those without fire emissions). These are plotted in Figure 7 Figure 8, while the ΔO₃ values can be found in Table 3. In Figure 7, we compare OLS MLR (left) to KRR (right), with the red triangles indicating the training datasets, the green circle as a single validation data point, and the black dots as an emulation over a wide range of parameters. The vertical axis in Figure 7 is ΔO₃. Figure 8 plots the same data, but instead of ΔO₃ the vertical axis is the NOₓ scaling factor and the colors of each point are the ΔO₃ values. The main difference between OLS MLR and KRR is that the OLS MLR predicts a clearly linear system with values that are comparatively higher than the non-linear KRR regression. For the validation data point (green dot in Figure 7 Figure 8), in which STILT-ASP predicts a ΔO₃ = 0.3898 ppb, the OLS MLR predicts ΔO₃ = 0.4538 ppb and the KRR predicts ΔO₃ = 0.4812 ppb. As there is only one validation data point used in this memo, we feel it is important to not over-interpret this result, especially as we ran addition simulations to expand the parameter base as described in the next section.
3.4.2 Preliminary Conclusions

The results presented above represent a preliminary analysis of the capabilities of KRR emulation. With only eight data points within the 3D parameter space, we were unable to divide the data points into sufficiently sized training and testing data sets, which we would need to evaluate the performance of KRR compared to the OLS MLR. In addition, the limited range of O₃ mixing ratios for the STILT-ASP data points presents a case of “small numbers” and “small differences.” For instance, the largest difference in the STILT-ASP datasets (i.e. 0.86 ppb O₃ difference) between the “top left” and “bottom right” results in the OLS MLR emulation to simply extend this gradient out in the expanded emulated parameter space, such that the highest O₃ values are still in the top left and the lowest values are in the bottom right of Figure 6. While this may
accurately represent the expanded parameter space, it is difficult to conclude with certainty without additional STILT-ASP simulated data points.

Similarly, the KRR emulation estimates \( O_3 \) mixing ratios to be the highest within the original STILT-ASP “box” and smaller values (as low as 50 ppb) at the extreme outer edges of the expanded parameter space (Figure 6). Again, this may be an accurate representation of the expanded parameter space, but we need to include additional STILT-ASP points before we can confidently draw conclusions. Additionally, some of the expanded parameters are physically unrealistic (i.e. the PBL parameter scaled to 0.0). Finally, evaluating the statistical performance of the emulation methodology requires dividing the STILT-ASP data into adequately sized training and testing datasets.

Next, we conduct additional STILT-ASP simulations for a wider range of the three parameters included in this memo. Specifically, we cover additional points towards the outer edges of the “parameter boxes” seen in Figure 6 Figure 8. The full parameter space of interest would include \( \text{NO}_x \) scaling values of \([0.5, 0.75, 1, 1.25, 1.5, 1.75, 2.0] \), PBL scaling values of \([0, 0.25, 0.5, 0.75, 1.0, 1.25, 1.5] \), and \( O_3 \) deposition values of \([0.5, 0.75, 1, 1.25, 1.5, 1.75, 2.0] \), but this would require \( 7 \times 7 \times 7 = 343 \) simulations in total. We select a subset (after removing unphysical parameters such as a PBL scaling of 0) so that we can cover the target parameter space without the full suite of simulations. These additional simulations enable us to create larger training and testing datasets, which allows us to have a more complete statistical comparison of OLS MLR and KRR.
### 3.5 Expanded Parameter Space for STILT-ASP Simulations and Emulation

Table 5: Parameter values and ∆O₃ (ppb) for all 23 STILT-ASP simulations

<table>
<thead>
<tr>
<th>Phase</th>
<th>Simulation</th>
<th>NOₓ Scaling</th>
<th>PBL Scaling</th>
<th>O₃ Deposition Scaling</th>
<th>∆O₃</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Preliminary</td>
<td>BBB</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.236</td>
</tr>
<tr>
<td></td>
<td>MBB</td>
<td>1.5</td>
<td>1</td>
<td>1</td>
<td>0.478</td>
</tr>
<tr>
<td></td>
<td>BMB</td>
<td>1</td>
<td>0.5</td>
<td>1</td>
<td>0.398</td>
</tr>
<tr>
<td></td>
<td>BBM</td>
<td>1</td>
<td>1</td>
<td>1.5</td>
<td>0.235</td>
</tr>
<tr>
<td></td>
<td>MMB</td>
<td>1.5</td>
<td>0.5</td>
<td>1</td>
<td>0.705</td>
</tr>
<tr>
<td></td>
<td>MBM</td>
<td>1.5</td>
<td>1</td>
<td>1.5</td>
<td>0.477</td>
</tr>
<tr>
<td></td>
<td>BMM</td>
<td>1</td>
<td>0.5</td>
<td>1.5</td>
<td>0.398</td>
</tr>
<tr>
<td></td>
<td>MMM</td>
<td>1.5</td>
<td>0.5</td>
<td>1.5</td>
<td>0.703</td>
</tr>
<tr>
<td></td>
<td>CCC</td>
<td>1.25</td>
<td>0.75</td>
<td>1.25</td>
<td>0.390</td>
</tr>
<tr>
<td>Corners</td>
<td>LML</td>
<td>0.5</td>
<td>0.5</td>
<td>0.5</td>
<td>0.018</td>
</tr>
<tr>
<td></td>
<td>HML</td>
<td>2</td>
<td>0.5</td>
<td>0.5</td>
<td>0.906</td>
</tr>
<tr>
<td></td>
<td>LHL</td>
<td>0.5</td>
<td>1.5</td>
<td>0.5</td>
<td>-0.112</td>
</tr>
<tr>
<td></td>
<td>LMH</td>
<td>0.5</td>
<td>0.5</td>
<td>2</td>
<td>0.016</td>
</tr>
<tr>
<td></td>
<td>LHH</td>
<td>0.5</td>
<td>1.5</td>
<td>2</td>
<td>-0.112</td>
</tr>
<tr>
<td></td>
<td>HHL</td>
<td>2</td>
<td>1.5</td>
<td>0.5</td>
<td>0.678</td>
</tr>
<tr>
<td></td>
<td>HMH</td>
<td>2</td>
<td>0.5</td>
<td>2</td>
<td>0.897</td>
</tr>
<tr>
<td></td>
<td>HHH</td>
<td>2</td>
<td>1.5</td>
<td>2</td>
<td>0.672</td>
</tr>
<tr>
<td>Centroids</td>
<td>CBH</td>
<td>1.25</td>
<td>1</td>
<td>2</td>
<td>0.360</td>
</tr>
<tr>
<td></td>
<td>CBL</td>
<td>1.25</td>
<td>1</td>
<td>0.5</td>
<td>0.365</td>
</tr>
<tr>
<td></td>
<td>CHC</td>
<td>1.25</td>
<td>1.5</td>
<td>1.25</td>
<td>0.334</td>
</tr>
<tr>
<td></td>
<td>CMC</td>
<td>1.25</td>
<td>0.5</td>
<td>1.25</td>
<td>0.563</td>
</tr>
<tr>
<td></td>
<td>HBC</td>
<td>2</td>
<td>1</td>
<td>1.25</td>
<td>0.670</td>
</tr>
<tr>
<td></td>
<td>LBC</td>
<td>0.5</td>
<td>1</td>
<td>1.25</td>
<td>-0.051</td>
</tr>
</tbody>
</table>

Fourteen additional STILT-ASP simulations were conducted, which, in addition to the 9 from the preliminary work presented above, results in a total of twenty-three STILT-ASP simulations that were conducted varying three parameters: (1) the magnitude of the NOₓ emissions; (2) the planetary boundary layer mixing height; and (3) the O₃ deposition velocity. A fourth parameter, the plume rise height, was modified but O₃ concentrations showed no sensitivity to this parameter; although there were slight differences in the distribution of pollutants in the vertical, they were small, so we left that parameter out of the sensitivity simulations. Table 5 summarizes all twenty-three simulations and their final ∆O₃ values. The abbreviations for the simulation names
are made up of three characters, each of which matches the three parameters in order (i.e. the NO$_x$ parameter, the boundary layer parameter, and the O$_3$ deposition parameter), with “B” meaning baseline, “M” meaning modified (for the preliminary simulations), “C” meaning centroid, “H” meaning high values, and “L” meaning low values. Thus Simulation “BBB” is the baseline simulation with no changed parameters, while “LBC” uses the low value for the NO$_x$ parameter, the baseline value for the boundary layer parameter, and the centroid value for the O$_3$ deposition parameter.

Figure 9 plots the different phases of simulations, starting with the preliminary phase conducted for the previous section (black box, top left of Figure 9). We expanded this box outwards for all the parameters (with the exception of expanding the planetary boundary layer parameter down to zero), as represented by the red box in the top right of Figure 9). Finally, we expanded to the “faces” or “centroids” of this expanded box, as represented in the blue cross in the lower left of Figure 9. The figure in the lower right of Figure 9 plots all three phases together. This set of points will be plotted in additional figures below but the black, red, and blue lines will not be. In the figures that use the three parameters, this shape and relationships for these points will be maintained, but in the figures below that select two parameters and then ΔO$_3$ as a third parameter, the size and shape of this set of points will be distorted as those plots are effectively projecting this three-dimensional parameter space into another parameter space.

![Figure 9: STILT-ASP simulated parameter space](image-url)
Figure 10: STILT-ASP ΔO₃ for full simulated parameter space

Figure 10 plots the ΔO₃ (colors) for the same points in the same parameter space as in Figure 9. Values for each of these points can also be found in Table 5. As can be seen, the highest ΔO₃ happens in the “top left” of Figure 10 reaching nearly to 1 ppb while the smallest ΔO₃ happens in the “bottom right” of Figure 10, reaching values below zero. Two-thirds of these points are randomly selected to serve as training data points for the emulation techniques while one-third of these points are reserved for validation and statistical evaluation. We developed two emulators, one being the KRR emulator as described in Section 3.2 and one being a more traditional multiple linear regression (MLR) technique, which optimized its fit using the ordinary least squares (OLS) metric.

3.6 KRR and MLR Results

Figure 11: KRR (left) and MLR (right) emulations for expanded parameter space
Figure 11 plots emulations of STILT-ASP using both KRR (left) and MLR (right). The STILT-ASP datapoints are plotted as large spheres, while a $10 \times 10 \times 10 = 1000$ emulated parameter space is plotted with the small spheres. We extended the 1000 emulated to points to fill the expanded “cube” plotted in Figure 10 Figure 11, as well as extended down to a planetary boundary layer scaling of zero. The two emulation techniques have comparable results, with subtle differences that can be most easily be seen in Figure 11 in the “upper left” corner where, while the MLR technique has the highest $\Delta O_3$ values in the “upper back left” corner, the KRR technique has maximum $\Delta O_3$ values more towards the center of the expanded parameter space. These differences are seen more clearly when we plot the results in different projections as we do in the following figures.

Figure 12: 3D reprojections of expanded parameter space from Figure 3.

Figure 12 plots the KRR (top row) and MLR (bottom row) results for different combinations of the three parameters (the NO$_x$ and PBL parameter in the left column, the O$_3$ deposition and NO$_x$ parameter in the center column, and the O$_3$ deposition and PBL parameter in the right column). The green spheres represent the STILT-ASP data points, which get “stretched” and “twisted.” It is clearly seen that both the KRR and MLR techniques emulate STILT-ASP similarly (i.e. a similar shape and span of the extended parameters plotted with black dots), with the KRR technique demonstrating more “curve” as it has capabilities to emulate non-linear features. The MLR technique cannot emulate non-linear features and thus its emulation results are linear. Interestingly, as the KRR technique weights more favorably in regions where there are more datapoints, it “misses” some values (i.e. the two isolated green spheres in the top right subplot of
Figure 12) as it has “pulled” points more towards the densely packed “preliminary” STILT-ASP training points.

Figure 13: 2D “top-down” reprojection of STILT-ASP emulation

Figure 13 plots the same data as in Figure 12 but over two dimensions at a time with colors indicating the relative values of the $\Delta O_3$ in ppb. In essence, each subplot in Figure 13 is “looking down from above” a corresponding plot in Figure 12, and thus the $\Delta O_3$ colors are only representative of the “top layer” of datapoints. We can see here that the NO$_x$ parameter (left and center column in Figure 13) has a clear impact on the $\Delta O_3$ (colors), with higher NO$_x$ emissions resulting in higher $\Delta O_3$ values (hot colors in Figure 13. In contrast, the PBL and O$_3$ deposition parameters have much less impact on the $\Delta O_3$ mixing ratios. Slight differences can be noted between the KRR emulation results (top row) and MLR emulation results (bottom row), and are similar to the features noted above, namely that the MLR results are clearly linear while the KRR results show some small deviations from linearity.

Finally, we plot in Figure 14 Figure 15 from one last perspective. Instead of looking “down from above” as in Figure 13, Figure 14 Figure 15 look at each of the subplots in Figure 12 “from the side,” with the vertical axis representing the $\Delta O_3$ values and the horizontal axis and the color scales representing different combinations of the three parameters. Figure 14 plots the KRR emulation results while Figure 15 plots the MLR emulation results. These plots demonstrate more clearly the two major findings: (1) the NO$_x$ emissions parameter has the largest impact on the $\Delta O_3$ values; and (2) that the KRR technique, while similar to the MLR technique overall, clearly demonstrates non-linear features that the MLR technique cannot.
Statistically, both the KRR and MLR techniques are comparable over the parameter space explored here, with $R^2$ values of 0.922 and 0.955 and RMSE values of 0.0114 and 0.00659 for KRR and MLR, respectively. Thus, there is no clear advantage over using either KRR or MLR techniques here, but future exploration of additional parameters are likely to have non-linear impacts on $\Delta O_3$, and thus we believe that the KRR technique would perform better than the MLR technique.

Figure 14: 2D “from the side” reprojections of STILT-ASP for KRR emulation
Figure 15: 2D “from the side” reprojections of STILT-ASP for MLR emulation
4 Complex Step Approach and Emission Uncertainty Study

4.1 Introduction

The purpose of this task was to implement the complex step approach in STILT-ASP for computing model output sensitivities to model inputs. We followed the methods of Constantin and Barrett (2014a) where they implemented the approach in the GEOS-Chem model. We have edited the source code of STILT-ASP to successfully incorporate the complex step approach in our analysis of $O_3$ and $PM_{2.5}$ sensitivity to anthropogenic, biogenic, and biomass burning emissions. We used the STILT-ASP model adapted with an improved plumerise and mixing parameterization as our base model (Section 2) and applied it to the University of Texas El Paso fire event on June 21, 2015.

4.2 Theoretical Basis of the Complex Step Approach

One of the key parameters needed from photochemical models is the sensitivity of model outputs, like $O_3$ mixing ratio, to model inputs, such as emissions from anthropogenic, biogenic, and biomass burning sources. Typically, the “finite difference” approach (Eq. 1) is used, which estimates model output sensitivity based on the difference in output between a perturbed and non-perturbed simulation (Constantin and Barrett, 2014b).

$$F'(x_0) = \frac{F(x_0 + \Delta) - F(x_0)}{\Delta}$$

The finite difference approach, while straightforward to implement, has limited applications for assessing model sensitivity at the scales required for realistic air quality applications. The limitations of the finite difference approach arise from the incurred truncation errors and cancellation errors that are non-negligible relative to the applied perturbation. As the value of $\Delta$ decreases, the truncation error decreases but is increasingly offset by an increase in cancellation error resulting from computational finite precision. As such, the uncertainty associated with the finite difference approach is not appropriate for important policy-level applications of air quality modeling—for example, analysis of the downstream effects of marginal perturbations of emissions at a single location in time (Constantin and Barrett, 2014b).

To circumvent the limitations of the finite difference approach, photochemical models like CAMx calculate first-order sensitivities using the Direct Decoupled Method (DDM; Koo et al., 2007). However, analytical differentiation schemes like DDM can take a large amount of time to develop and to maintain as the CTM code evolves. An alternative approach to calculating sensitivities is called the complex step approach (Squire and Trapp, 1998; Martins et al., 2003; Constantin and Barrett, 2014b). Lyness and Moler (1967) first exploited complex numbers to calculate sensitivities based on Cauchy’s theorem when analytical derivatives were hard to develop. Squire and Trapp (1998) reframed their technique as a variation of the finite difference approximation of the first derivative (Eq. 2), which is the basis of modern applications of this technique.

$$\frac{dF(x_0)}{dx} \approx \frac{F(x_0 + ih_p) - F(x_0 - ih_p)}{2ih_p} = \frac{Im(F(x_0 + ih_p))}{h_p}$$

Equation 2 shows how the local model sensitivity to a perturbation, $h_p$, can be approximated with the imaginary component of the perturbed model output. The complex step approach
eliminates subtraction cancellation error by applying the perturbation, $h_p$, in imaginary space. Such small perturbations can be applied (e.g., $10^{-30}$) such that the truncation error, which is $\sim O(h_p^2)$, is negligible compared to the analytical accuracy of numerical solutions. Operator overloading can be used to limit the changes to the original source code to changing the variables of interest to complex type.

Constantin and Barrett (2014a) applied the complex step approach (XPLEX) to computing sensitivities in the GEOS-Chem chemistry transport model. They have provided a general XPLEX code that can be applied to any fortran-based chemistry transport model. Constantin and Barrett (2014a) also describe why the use of the intrinsic Fortran COMPLEX*16 data type is inappropriate for our chemistry transport modeling purposes.

In this task, we have ingested the XPLEX code into STILT-ASP and have used the XPLEX version of STILT-ASP to explore the sensitivity of $O_3$ and $PM_{2.5}$ to biomass burning emissions, as well as to anthropogenic and biogenic emissions, for cases previously studied with STILT-ASP by Brown-Steiner et al. (2018). We discuss the implementation details in Section 3 and preliminary results in Section 4.

4.3 Details of Complex Step Approach Implementation in STILT-ASP

We followed the general guidelines established by Constantin and Barrett (2014a) for implementation of the XPLEX code into STILT-ASP. We made source code changes to the version of STILT-ASP containing an improved biomass burning plume rise parameterization (Section 2).

4.3.1 Summary of XPLEX Procedure

As described in Constantin and Barrett (2014a), we followed the overall procedure for ingesting the XPLEX code into the STILT-ASP model (STILT-ASP_XPLEX):

i. Convert all single-precision (real) variables to double-precision (real*8).
ii. Convert all real*8 to type XPLEX by executing the developer-provided modifyx.plx perl script
iii. Copy the fortran modules mytype.f90 and xplexify.f90 to the local library
iv. Manually modify local library makefiles to incorporate build commands for mytype.f90 and xplexify.f90
v. Manually modify code read/write statements to account for the presence of complex-valued variables
vi. Apply a small complex step as a perturbation to desired input variable(s) and run model. In our STILT-ASP_XPLEX, these perturbations are currently applied by a manual switch in the hymodelc_forward.f90 script where one (or none for a base-state run) of the variables perturb_finn, perturb_camx, and perturb_megan is set to .TRUE. In future versions of the STILT-ASP_XPLEX, these perturbations will be read from a configuration file.

4.3.2 Modifications to XPLEX Code

We modified the xplexify.f90 code to

i. incorporate miscellaneous bug fixes, and
ii. expand the XPLEX suite of operations for XPLEX-type variables where no pre-existing rule was defined. These were mainly cases where arrays of different dimensionality had to
be multiplied, and the new routines were based on the routines in the original xplexify.f90 module.

No other changes were made to the XPLEX developer code.

4.3.3 STILT-ASP_XPLEX Troubleshooting

There were several instances where second-order customization and bug fixing of the XPLEX procedure were conducted prior to successfully running the STILT-ASP_XPLEX model. The issues and solutions are described below.

i. **Requirement of “IMPLICIT NONE” declarations.** The modify.plx script only works when the fortran implicit type-setting feature is disabled via the “IMPLICIT NONE” declaration before each function, module, or subroutine. Specifically, modify.plx inserts MYTYPE and XPLEXIFY use statements and replaces real*8 variables with TYPE (XPLEX) only where “IMPLICIT NONE” is declared. While it is recommended FORTRAN programming practice to always declare “IMPLICIT NONE”, this is not always the case, and several STILT library functions derived from HYSPLIT do not include this statement. In certain instances, we had to manually add “IMPLICIT NONE” in segments of the STILT-ASP code in order for modify.plx to take effect and for the compiler to correctly pass information between modules and subroutines.

ii. **Exceptions to real*8.** Declarations of type DATA and PARAMETER need to stay as real or real*8; they cannot be of type XPLEX.

iii. **Type-specific functions and type conversions.** Type-specific functions expect real-valued arguments and therefore need to be replaced with generic functions to allow for type XPLEX arguments. For example, the absolute value function DABS expects a real-valued input; we replace these instances with ABS which takes integer, real, or complex inputs as arguments. In cases where type conversions occur (for example, double() ), these had to be removed and replaced with the closest equivalent from the xplexify.f90 code.

iv. **Variable name conflicts.** In certain instances, variables were named with implicit function names (e.g., Sum, Min, Max). These names caused conflicts with the XPLEX code, and the variables had to be re-named. In another instance, ISNAN could not be defined in the source code as it was already defined by the compiler.

v. **Complex – XPLEX conflicts.** In the optical properties routine, functions inherently requiring the use of complex math conflicted with XPLEX type declarations. As aerosol optical properties were not a focus of this work, we fixed this issue by commenting out the relevant subroutines.

vi. **Single-valued variable assignments.** In read statements where the original real (or real*8) variable was set to a single value, the imaginary component of the XPLEX variable equivalent needed to be manually set to 0 while the real component only was set to the original value.

vii. **CAMx libraries.** The xplexify module expects that all reals will be double precision (real*8) as in GEOS-Chem. Most STILT-ASP code is automatically compiled with a -r8 (real*8) flag which prevents conflicts in instances where a real value is multiplied with an XPLEX type value. However, as the CAMx emission files are FORTRAN binary files, they must be read in using the same precision that they were written out with, in this case real*4. Thus, in the CAMx library, adding an -r8 compiler flag would have caused conflicts. In addition, our conversion of CAMx code to incorporate TYPE (XPLEX)
variables had to be manually edited to re-set certain variables to their original single-precision real status. To avoid “xplexifying” unused code, we also modified the CAMx makefile to compile only the CAMx code used by STILT-ASP. Previously, all CAMx code was compiled. In the STILT-ASP_XPLEX version only five essential scripts are compiled: area_source_emissions_mod.f90, point_source_emissions_mod.f90, scale_factor_modules_new.f90, camx_mapping_mod.f90, and lcpgeo.f90. Of these, only lcpgeo.f90 is from the CAMx model, the rest were written at AER for STILT-ASP.

viii. Development of a non-zero imaginary component. During the course of the various test model runs we noticed a small (<10^{-20}) growth of the imaginary components which should have been zero during the non-perturbed runs. We tracked down the sources to three broad categories. The imaginary components in these three cases are not relevant for the current scope of our sensitivity studies, and could therefore be hard-coded to “0”.

   a. Solar zenith angles, relevant to photolysis reactions. Chemistry.f90
   b. Gas phase chemistry reaction rate constants. ChemistryParametersInitRoutines.f90, Chemistry.f90, StepASP4STILT.f90
   c. Dry and wet deposition velocities. hymodelc_forward.f90.

ix. Inclusion of secondary PM_{2.5}. We turned off secondary aerosol chemistry while we were resolving the source of non-zero imaginaries. We have now successfully included secondary aerosol routines. We have not included aqueous organic aerosol condensation subroutines as this was causing errors in the ODE calculations. The hydrophobic nature of organic aerosols makes the effect of excluding the aqueous organic aerosol condensation subroutine negligible for our purposes.

x. Concentration differences between non-XPLEX and base XPLEX runs. We fixed a bug in a particle location calculation in the XPLEX version of the code. The bug resulted in an error in the solar zenith angle calculation, which in turn significantly impacted modeled concentrations of species like O_3 whose evolution relies on photochemistry.

4.3.4 Sensitivity Analysis: Preliminary Results from Complex Step Approach

We ran five versions of the STILT-ASP model and these results are displayed in Table 6. We ran a non-XPLEX case (“Original”) based on the STILT-ASP non-XPLEX plumerise version (Section 2); a non-perturbed base case (“Base”); and then individually applied a small perturbation, \( h_p=10^{-6} \) for one of each of biomass burning emissions (FINN), anthropogenic emissions (CAMx), or biogenic emissions (MEGAN). Table 6 displays the sensitivity of various species output to these perturbations (as defined in the last term of Eq. 2). The reported species of interest are O_3, PM_{2.5}, CO, NO, NO_2, and PAN.

The sensitivity provides an estimate of the effects of an increase of emissions, assuming a linear system response. Equation 3 illustrates this relationship using O_3 sensitivity (\( \Delta O_3 \)) to a change in fire emissions (\( E_{fire} \)). These sensitivities are the first-order derivative of the species concentration with respect to the scale factor for the emissions. Thus, to estimate the change in concentration to a 10% change in emissions, this sensitivity would be multiplied by 0.1.

\[
\frac{dO_3}{dE_{fire}} \times \Delta E_{fire} = \Delta O_3
\]
We can also assess the relative importance of emission type to each species for an equivalent emissions perturbation. For example, Table 6 shows that O$_3$ is most sensitive to perturbations in anthropogenic and biogenic emissions. Relative to anthropogenic emissions, O$_3$ sensitivity to equivalent perturbations of biogenic and fire emissions is lower by 20% and 88%, respectively. On the other hand, PM$_{2.5}$ is most sensitive to fire emissions. For an equivalent perturbation, PM$_{2.5}$ sensitivity to fire emissions is greater than anthropogenic emissions and biogenic emissions by a factor of about 11 and 7, respectively.

Table 6: Summary of results from perturbed and non-perturbed STILT-ASP_XPLEX simulations. “r” and “i” refer to real and imaginary components respectively. *Refers to STILT-ASPv2.0 modified to include the plumerise algorithm from WRF-Chem (Dayalu & Alvarado, 2019).

<table>
<thead>
<tr>
<th></th>
<th>*Original: v2.0</th>
<th>Base: vXPLEX</th>
<th>Fires: FINN</th>
<th>Anthropogenic: CAMx</th>
<th>Biogenic: MEGAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h_p$</td>
<td>N/A</td>
<td>--</td>
<td>1E-06</td>
<td>1E-06</td>
<td>1E-06</td>
</tr>
<tr>
<td>$rO_3_{out}$ (ppb)</td>
<td>81.68</td>
<td>81.69</td>
<td>81.69</td>
<td>81.69</td>
<td>81.69</td>
</tr>
<tr>
<td>O$<em>3$ Sensitivity ($iO_3</em>{out} h_p^{-1}$)</td>
<td>N/A</td>
<td>--</td>
<td>0.2895</td>
<td>2.508</td>
<td>2.000</td>
</tr>
<tr>
<td>$rPM_{2.5, out}$ ($\mu g m^{-3}$)</td>
<td>13.85</td>
<td>14.18</td>
<td>14.18</td>
<td>14.18</td>
<td>14.18</td>
</tr>
<tr>
<td>PM$<em>{2.5}$ Sensitivity ($iPM</em>{2.5, out} h_p^{-1}$)</td>
<td>N/A</td>
<td>--</td>
<td>1.770</td>
<td>0.1523</td>
<td>0.2291</td>
</tr>
<tr>
<td>$rCO_{out}$ (ppb)</td>
<td>146.6</td>
<td>146.6</td>
<td>146.6</td>
<td>146.6</td>
<td>146.6</td>
</tr>
<tr>
<td>CO Sensitivity ($iCO_{out} h_p^{-1}$)</td>
<td>N/A</td>
<td>--</td>
<td>1.842</td>
<td>7.252</td>
<td>2.495</td>
</tr>
<tr>
<td>$rNO_{out}$ (ppb)</td>
<td>0.2098</td>
<td>0.2086</td>
<td>0.2086</td>
<td>0.2086</td>
<td>0.2086</td>
</tr>
<tr>
<td>NO Sensitivity ($iNO_{out} h_p^{-1}$)</td>
<td>N/A</td>
<td>--</td>
<td>-0.4020E-02</td>
<td>0.1256</td>
<td>-0.3023</td>
</tr>
<tr>
<td>$rNO_{2, out}$ (ppb)</td>
<td>0.6764</td>
<td>0.6728</td>
<td>0.6728</td>
<td>0.6728</td>
<td>0.6728</td>
</tr>
<tr>
<td>NO$<em>2$ Sensitivity ($iNO</em>{2, out} h_p^{-1}$)</td>
<td>N/A</td>
<td>--</td>
<td>-0.1022E-01</td>
<td>0.3479</td>
<td>-0.5204E-01</td>
</tr>
<tr>
<td>$rPAN_{out}$ (ppb)</td>
<td>0.2047</td>
<td>0.2054</td>
<td>0.2054</td>
<td>0.2054</td>
<td>0.2054</td>
</tr>
<tr>
<td>PAN Sensitivity ($iPAN_{out} h_p^{-1}$)</td>
<td>N/A</td>
<td>--</td>
<td>0.1579E-02</td>
<td>0.5806E-01</td>
<td>0.9719E-01</td>
</tr>
</tbody>
</table>

We conducted a test to ensure that the results from the base (non-perturbed) run of STILT-ASP_XPLEX match the non-XPLEX version of the model. We find small differences (<0.01% to 2%; see Table 6) between the two runs that require further exploration. The impacts of (a) are less important, as they are typically $O(10^{-5}$ to $10^{-8}$). Figures 16-20 display particle*timestep evolution over the entire simulation for the key species (except PM2.5, which has multiple precursors) from Table 6. Figures 16-20 demonstrate that the residuals between the non-XPLEX and XPLEX base runs are small but distinctly non-random in several areas, consistent among the key species. We therefore hypothesize the issue may be associated with a combination of the following: (a) Fortran padding real*8 variables with random digits upon conversion from real*4 which would result in a
random distribution of non-XPLEX and XPLEX residuals; (b) complications with the differential equation solver in the XPLEX version of the code; and (c) bugs in the XPLEX photochemistry code. The greatest discrepancy is in the PM$_{2.5}$ output (XPLEX base run is 2.4% higher than non-XPLEX). Given the nature of the small but non-random errors displayed in Figures 16-20 (c,d) we hypothesize that the effects of these errors are aggregating across the multiple PM$_{2.5}$ precursors (including those not listed in Table 6 or displayed in Figures 16-20), leading to the relatively larger 2.4% discrepancy.

Figure 16: $\Delta$O$_3$ for each of 500 particles at each 15-minute timestep of base case STILT-XPLEX and NONXPLEX 48h forward model simulations (96000 points). (a) Comparison of NONXPLEX (black) and XPLEX (red) along 48h of the simulation; XPLEX point size reduced for visual clarity (b) comparison of XPLEX vs. non-XPLEX with 1:1 line for comparison, (c) NONXPLEX-XPLEX residuals, and (d) histogram of residuals.
Figure 17: ΔCO for each of 500 particles at each 15-minute timestep of base case STILT-XPLEX and NONXPLEX 48h forward model simulations (96000 points). (a) Comparison of NONXPLEX (black) and XPLEX (red) along 48h of the simulation; XPLEX point size reduced for visual clarity (b) comparison of XPLEX vs. non-XPLEX with 1:1 line for comparison, (c) NONXPLEX-XPLEX residuals, and (d) histogram of residuals.
Figure 18: ΔNO for each of 500 particles at each 15-minute timestep of base case STILT-XPLEX and NONXPLEX 48h forward model simulations (96000 points). (a) Comparison of NONXPLEX (black) and XPLEX (red) along 48h of the simulation; XPLEX point size reduced for visual clarity (b) comparison of XPLEX vs. non-XPLEX with 1:1 line for comparison, (c) NONXPLEX-XPLEX residuals, and (d) histogram of residuals.
Figure 19: $\Delta$NO$_2$ for each of 500 particles at each 15-minute timestep of base case STILT-XPLEX and NONXPLEX 48h forward model simulations (96000 points). (a) Comparison of NONXPLEX (black) and XPLEX (red) along 48h of the simulation; XPLEX point size reduced for visual clarity (b) comparison of XPLEX vs. non-XPLEX with 1:1 line for comparison, (c) NONXPLEX-XPLEX residuals, and (d) histogram of residuals.
Figure 20: ΔPAN for each of 500 particles at each 15-minute timestep of base case STILT-XPLEX and NONXPLEX 48h forward model simulations (96000 points). (a) Comparison of NONXPLEX (black) and XPLEX (red) along 48h of the simulation; XPLEX point size reduced for visual clarity (b) comparison of XPLEX vs. non-XPLEX with 1:1 line for comparison, (c) NONXPLEX-XPLEX residuals, and (d) histogram of residuals.
5 Quality Assurance

All model source code, scripts, and output files were examined by a team member who did not participate in their creation. This review satisfies the requirement to review 10% of the data from the QAPP. As the plume rise calculation did not result in significant changes to the STILT-ASP predicted concentrations, additional evaluations of model skill were not required. Our current QA focus is on resolving the final concentration differences between the original and complex-step approach versions of STILT-ASP noted in Section 4. Assuming those can be resolved in time for the final report, we will perform additional evaluations of both models and include the results in the final report.

In addition, the QAPP listed the following questions to be answered during the quality assessment:

- **How well does the updated plume rise scheme reflect the BBOP observations of plume heights?**
  The plume rise calculation in general did not result in significant changes to the concentration predictions of the STILT-ASP models, suggesting that most plume heights were determined to be under 2000 m in altitude, the assumed minimum mixing height in STLT-ASP. Thus, no further comparisons with BBOP or other data were performed.

- **How does the mixing parameterization improve or degrade the ability of STILT-ASP to reproduce the HGB observations during DISCOVER-AQ?**
  As already noted, the plume rise calculation in general did not result in significant changes to the concentration predictions of the STILT-ASP models, so the evaluations versus HGB during DISCOVER-AQ were unchanged.

- **Under what conditions is the updated model expected to be valid?**
  The model is expected to be valid for 5-day back-trajectories within North America but primarily for urban-average concentrations due to the low spatial resolution of the meteorological and emission input fields used.

- **Do the changes in the model predictions of the impact of fires and other sources on O\textsubscript{3} and PM\textsubscript{2.5} due to changes in the uncertain parameters fit with our conceptual model of the effect of these changes?**
  The changes in O\textsubscript{3} and PM\textsubscript{2.5} due to perturbations of O\textsubscript{3} deposition, minimum mixing height, NO\textsubscript{x}/VOC ratios in the fire emissions, and plume injection height are generally consistent with the expected impact of uncertainties in these variables. However, we were surprised at how little impact changes to the assumed heat flux and fire size in the plume rise algorithm had on the predicted plume injection height.

- **Under what conditions is the KRR emulator expected to be valid?**
  The KRR emulator developed in this work is only expected to be valid for the specific case simulated and within the range of parameters that it was trained on. Its main purpose was to identify which of the uncertain parameters tested had the largest net impact on the model results, but it should not be used as a replacement for the full STILT-ASP model.
• *Do the results of the uncertainty study suggest any clear targets for reducing the uncertainty in the STILT-ASP model predictions?*

Our results suggest that the parameters examined in this study -- NOx emissions, boundary layer height, O3 deposition, and fire size -- impact the contribution of fires to O3 only moderately (< 1 ppb). Thus, other targets should be focused on for further improvements, such as increasing the resolution of the emissions and meteorology, improvements in the FINN fire emissions, and improvements in the chemistry in ASP.

• *Do the results of the KRR emulator evaluation suggest any ranges of uncertain variables are not consistent with the observations?*

While our evaluation suggests that KRR is consistent with the ranges of the uncertain variables it was trained on, for these cases it was not significantly better at predicting concentrations in areas where it wasn’t trained than a multiple linear regression approach.

• *Is the complex-step version of the code consistent with the finite-difference results of the original STILT-ASP code?*

As of this final report, the concentrations from the base complex step STILT-ASP version slightly vary (<0.01% to 2%, species dependent) from the original source code. As the differences are very small but also exhibit distinctly non-random behavior in certain areas, we believe the error is associated with both a combination of fortran type-conversion padding and bugs in the “xplexified” differential equation solver. This will be a subject for ongoing XPLEX model development (Section 4.3).

• *Are the uncertainties in O3 and PM2.5 predictions due to emissions consistent with our conceptual models of O3 and PM2.5 emissions, chemistry, and transport during these events?*

Yes, the sensitivities of O3 and PM2.5 due to changes in fire, anthropogenic, and biogenic emissions are consistent with our conceptual models of the impact of these emissions.
6 Conclusions

Here we summarize the conclusions of our project, with reference to the corresponding report section.

- We successfully integrated a plume rise algorithm into STILT-ASP based on that used within WRF-Chem. However, this had little effect on the simulated concentrations in our test simulations, with most of the plume injections heights calculated as well below our minimum mixing height of 2000 m (Section 2).
- We conducted 23 simulations of STILT-ASP for a single location in El Paso, Texas varying the scaling factors for three variables: the NOx emissions, the planetary boundary layer height, and the O3 deposition rate. We also tested a fourth parameter, the size of the biomass burning fire, but although this slightly modified the height to which the plumes would rise, it did not impact the ΔO3 (Section 3.3).
- We developed two emulators for these 23 STILT-ASP simulations trained on the three variables towards predicting the O3 enhancement from wildfires, ΔO3, one using the traditional ordinary least squares multiple linear regression (OLS MLR) and one using kernel ridge regression (KRR). KRR is a type of regression that is capable of capturing non-linear features, such as the well-known non-linearity of O3 in the troposphere (Section 3.5).
- Overall, we found that ΔO3 in the STILT-ASP simulations was not very sensitive to these parameters, with the final ΔO3 ranging from -0.1 to +0.9 ppb. In addition, over this parameter space, the ΔO3 didn’t demonstrate significant nonlinearity. As such, both the OLS MLR and KRR emulation techniques proved equally capably of emulating the STILT-ASP results for these conditions and this set of parameters. However, the KRR emulation clearly demonstrated a degree on non-linearity that OLS MLR did not (Section 3.6).
- We have developed a complex-step sensitivity version of the STILT-ASP model. This process was more labor-intensive than anticipated due to the use of non-Fortran 90 compliant Fortran code in STILT. However, we were able to resolve these issues and give tips for how to address them for other legacy Fortran codes (Section 4.3)
- As of this final report, the concentrations from the complex step STILT-ASP version slightly vary (<0.01% to 2%, species dependent) from the original source code. As the differences are very small but also exhibit distinctly non-random behavior in certain areas, we believe the error is associated with both a combination of fortran type-conversion padding and bugs in the “xplexified” differential equation solver. This will be a subject for ongoing XPLEX model development (Section 4.3).
- The sensitivities of O3, PM2.5, and other species relative to fire, anthropogenic, and biogenic emissions calculated using the complex-step approach are consistent with our expectations. O3 is most sensitive to perturbations in anthropogenic and biogenic emissions. Relative to anthropogenic emissions, O3 sensitivity to equivalent perturbations of biogenic and fire emissions is lower by 20% and 88%, respectively. On the other hand, PM2.5 is most sensitive to fire emissions (Section 4.3).
7 Recommendations for Further Study

Based on the results of this work, we make the following recommendations for further study:

- Improving injection heights in STILT-ASP will likely require an improvement in plumerise algorithms beyond those available within WRF-Chem and used in this work. While multiangle passive sensors and lidars can give estimates of plume heights, one of the key limitations of these approaches is that identifying the plumes usually is a labor intensive process. New techniques to automatically identify smoke plumes would greatly help to advance our understanding of plume rise.
- Future work developing STILT-ASP should focus on other uncertain parameters such as increasing the resolution of the emissions and meteorology, improvements in the FINN fire emissions, and improvements in the chemistry in ASP.
- Future work could also explore the emulator’s capability to estimate STILT-ASP output for different sites and at different times of year.
- While the KRR and OLS MLR regressions were comparable for this site, we expect, due to the highly nonlinear nature of tropospheric O3 chemistry, that the ability of KRR to capture non-linear features indicates that KRR emulation would outperform OLS MLR regression under a wider set of parameters.
- The complex-step version of STILT-ASP should be used to investigate the sensitivity of the prediction fire O3 contribution to other uncertain variables that are usually hard to perturb in 3D photochemical simulations, such as meteorological variables.
8 References


Appendix A: Changes to STILT-ASP Source Code for Plumerise Algorithm

1. **Overview of modules/scripts added/edited.** We made changes to the STILT-ASP container_v2.0 obtained from the GitLab_Lexington repository.

```
GitSTILT-ASP#: git status
# On branch container_v2.0
# Changes not staged for commit:
# (use "git add <file>..." to update what will be committed)
# (use "git checkout -- <file>..." to discard changes in working directory)
# modified:   FINN_PREPROC/finn_lib.py
# modified:   FINN_PREPROC/htap_lib.py
# modified:   FINN_PREPROC/megan_lib.py
# modified:   FINN_PREPROC/stilt_asp_emissions_preprocessor.py
# modified:   Libraries/Dependencies/python_scripts/Plot2DOzone_v3a_loop2.py
# modified:   Libraries/Dependencies/python_scripts/plotBasemap2.py
# modified:   Libraries/Dependencies/python_scripts/stiltasp_parcel_analysis2.py
# modified:   Libraries/STILT-Chem/includes/CB4CHEMBK.INC
# modified:   Libraries/STILT-Chem/src/CB4METV.f
# modified:   Libraries/STILT-Chem/src/read_particle_netcdf.f90
# modified:   Libraries/STILT-Chem/src/STILT-Chem.f
# modified:   Libraries/STILT-Chem/src/module_chem_plumerise_scalar.f90
# modified:   Libraries/STILT-Chem/src/module_model_constants.f90
# modified:   Libraries/STILT-Chem/src/module_zero_plumegen_coms.f90
# modified:   stiltasp_wrapper.py
# Untracked files:
# (use "git add <file>..." to include in what will be committed)
# Libraries/STILT-Chem/src/module_chem_plumerise_scalar.f90
# Libraries/STILT-Chem/src/module_model_constants.f90
# Libraries/STILT-Chem/src/module_zero_plumegen_coms.f90
```

2. **Detailed description of changes by script/module, from (1) above and grouped according to similarity of change.** *In actual code, all changed instances are commented with a “plumerise” keyword*

**i. PYTHON SCRIPTS I: Fix basemap errors in pre- and post-processing scripts**

```
# modified:   Libraries/Dependencies/python_scripts/Plot2DOzone_v3a_loop2.py
# modified:   Libraries/Dependencies/python_scripts/plotBasemap2.py
# modified:   Libraries/Dependencies/python_scripts/stiltasp_parcel_analysis2.py
# modified:   FINN_PREPROC/htap_lib.py
# modified:   FINN_PREPROC/megan_lib.py
```

These python scripts were modified to include a fix to enable Basemap to work without causing the model build to crash. Basemap is in the process of being phased out, and is beginning to have dependency issues. In future, switching to the python package cartopy will more robustly resolve this issue. In the interim however, we simply incorporated the following line of code directly after the “import os” call to ensure that Basemap continues to work for the purposes of the STILT-ASP model build (and post-processing.)

```
import os
os.environ[‘PROJ_LIB’] = ‘/STILT-ASP/miniconda2/share/proj’
```
### ii. PYTHON SCRIPTS II: Incorporate new fire variables in FINN preprocessor

<table>
<thead>
<tr>
<th># modified: FINN_PREPROC/finn_lib.py</th>
</tr>
</thead>
<tbody>
<tr>
<td># modified: FINN_PREPROC/stilt_asp_emissions_preprocessor.py</td>
</tr>
</tbody>
</table>

These python scripts were modified to incorporate necessary fire variables into the FINN fire netcdf file created from the FINN Global fire file (e.g., GLOBAL_FINNv15_2015_MOZ4_7112014.txt). The variables are ultimately ingested by the plumerise algorithm added to the hymodelc_forward.F90 program. Note: any existing processed FINN netcdf file must be removed for the preprocessor to create a new one.

- **stilt_asp_emissions_preprocessor.py**: Define FIRETYPE, FIRESIZE, MEANFCT variables

  ```python
  fire_arr = ('FIRESIZE', 'MEANFCT')
  
  # set species variables and attributes
  for num in range(0, len(fire_arr)):
    varname = str(fire_arr[num]).strip()
    var = rootgrp.createVariable(varname, 'f4', ('FIRETYPE', 'HOUR', 'ROW', 'COL'))
    var.setncattr('long_name', varname)
    if varname == 'FIRESIZE':
      var.setncattr('units', 'm2')
      var.setncattr('var_desc', 'fire size by type')
    else:
      var.setncattr('var_desc', 'fraction of total fire size by type')
      var.setncattr('units', 'fraction')

  for iday in range(0, ndays):
    # Open intermediate netCDF file for this day
    intermed_file = intermed_path + '/'+prefix+'_modeldomain_soa_' + str(year) + '_' + str(idays[iday])+'_.'+'.nc'
    try:
      fhand = Dataset(intermed_file, 'r', format='NETCDF4')
      except Exception as excp:
        logging.error('Could not open intermediate FINN NetCDF file %s.'+\
                        ' Exception was %s', intermed_file, repr(excp))
        sys.exit()
    try:
      tmp = fhand.variables[varname][:]
    except Exception as excp:
      logging.error('Could not find variable %s in file %s.'+\
                        ' Exception was %s', varname, intermed_file, repr(excp))
      sys.exit()

    # Concatenate arrays along time dimension
    if iday == 0:
      out = tmp
    else:
      out = np.concatenate((out, tmp), axis=1) # MJA Bug fix 4/2019

    # Close the file
    fhand.close()

  num_vars = len(spec_arr)+4 # Total number of variables, species plus
  ```
• finn_lib.py (Converts raw annual FINN emission files in CSV format to NetCDF files matching a given lat-lon grid from a WRF GRIB file.)

```python
#lat and lon and firesize and meanfct
rootgrp.setncattr('NVARS', num_vars)
rootgrp.setncattr('NSPECS', num_vars-4) #Total number of species

#Sum fire sizes for each plume rise type (MJA, 20190325)
print finn_line[2]
if (float(finn_line[2]) > 3.9 and float(finn_line[2]) < 6.1): #temperate forest
    fire_size[0,0,pointeri, pointerj] = fire_size[0,0,pointeri, pointerj] +
    float(finn_line[5]) #m2
elif (float(finn_line[2]) > 2.9 and float(finn_line[2]) < 3.1): #tropical
    fire_size[1,0,pointeri, pointerj] = fire_size[1,0,pointeri, pointerj] +
    float(finn_line[5]) #m2
else: #grasslands/croplands
    fire_size[3,0,pointeri, pointerj] = fire_size[3,0,pointeri, pointerj] +
    float(finn_line[5]) #m2

#If sum of all fire sizes is greater than 2 km2, renormalize to 2 km2. (MJA, 20190325)
s = np.sum(fire_size, axis = 0) #sum along types, get 2D Lat/lon array
for i in range(0, len(lats[:, 0])):
    for j in range(0, len(lats[0, 1])):
        if sum_fire_size[0,i,j] > 0.0:
            for k in range(0, hour_len):
                fire_size[:,k,i,j] = fire_size[:,0,i,j]
                mean_fct[:,k,i,j] = fire_size[:,0,i,j]/sum_fire_size[0,i,j]
                if (sum_fire_size[0,i,j] > 2.0e6):
                    fire_size[:,k,i,j]=
                    2.0e6*(fire_size[:,0,i,j]/sum_fire_size[0,i,j])
mfct = rootgrp.createVariable('MEANFCT', 'f4', ('FIRETYPE','HOUR','ROW',
'COL'))
mfct[:] = mean_fct[:]
mfct.units = 'fraction of total fire area from this type'
fs = rootgrp.createVariable('FIRESIZE', 'f4', ('FIRETYPE','HOUR','ROW','COL'))
fs[:] = fire_size[:,:]
fs.units = 'square meters'
```

### iii. FORTRAN 90 PLUMERISE MODULES FROM WRF-Chem

| # Libraries/STILT-Chem/src/module_chem_plumerise_scalar.f90 |
| # Libraries/STILT-Chem/src/module_model_constants.f90 |
| # Libraries/STILT-Chem/src/module_zero_plumeigen_coms.f90 |

In the `module_zero_plumeigen_coms.f90` script, we adjusted the `nk` parameter to reflect the NARR gridbox dimensions of 28 levels. In the `module_chem_plumerise_scalar.f90` script we added the following:

```fortran
do ispc=1,nspecies
  eburn_out(1,ispc) = eburn_in(ispc)
  !AD PLUMERISE added in to prevent eburn_out from continuously adding to previous fire amount.
  do k=2,ml
    eburn_out(k,ispc) = 0
  enddo
  !!!!!
```
No other changes (other than print statements for debugging, later commented out) were made to the plumerise modules.

iv. FORTRAN 77/90 SUBROUTINES, INCLUDE FILES, MAKEFILES

| modified: Libraries/STILT-Chem/includes/CB4CHEMBK.INC |
| modified: Libraries/STILT-Chem/src/cb4metv.f |
| modified: Libraries/STILT/src/CB4CHEMBK.INC |
| modified: Libraries/STILT/src/cb4metv.f |
| modified: Libraries/STILT-Chem/src/read_particle_netcdf.f90 |
| modified: Libraries/STILT-Chem/src/Makefile |

Changes were made to the modules in this section to enable output of vertical profiles of \( u \) and \( v \) wind components (converted from grid/minute to m/s). The 3D output (particles, timesteps, levels) of \( u \) and \( v \) components (designated as “\( uuwind \)” and “\( vvwind \)” ) need to be added to the PARTICLE.nc file produced by the main hymodelc_backward.F90 program. This is then required by the call to the plumerise subroutines added to the hymodelc_forward.F90 program. Specific changes are:

- **cb4metv.f77 files (identical in both locations): calculate \( uuwind, vvwind \)**

```fortran
SUBROUTINE cb4metv(ip, jp, zp, kp, ii, pp, tgc, pgc, dgc, qgc, dsgc, qgcv, pu0, pv0, pt0 & 
                   , uuwind, vvwind)
real, intent(out) :: uuwind(:, :, :)
real, intent(out) :: vvwind(:, :, :)
uuwind(ii, pp, :) = u(ip, jp, :, kt, kg)*gx(ip, jp, kg)/60.0
vvwind(ii, pp, :) = v(ip, jp, :, kt, kg)*gx(ip, jp, kg)/60.0
```

- **CB4CHEMBK.INC include files (identical in both locations): register \( uuwind, vvwind \)**

```fortran
SUBROUTINE cb4metv(ip, jp, zp, kp, ii, pp, tgc, pgc, dgc, qgc, dsgc, qgcv, pu0, pv0, pt0 & 
                   , uuwind, vvwind)
real, intent(out) :: uuwind(:, :, :)
real, intent(out) :: vvwind(:, :, :)
```

- **read_particle_netcdf.f90 f90 file:**
  A. add MEANFCT, FIRESIZE, FIRETYPE definitions

```fortran
$Line 10:
allocate_real_4_4D

#Create a new data type:
type finn_variable2
!sequence
  character(len=64) :: firevar_name
  character(len=64) :: units2
  real, allocatable :: ffire(:, :, :, :)
end type

#Add new types to finn file structure:
type finn_file_struct
  integer :: firetype
  type(finn_variable), allocatable :: species(:)
  type(finn_variable2), allocatable :: ffvar(:)
end type

#Add in new subroutines for the two new variables of new type.
```
subroutine copy_finn_variable2(source2, destination2)
    implicit none

    type(finn_variable2), intent(in) :: source2
    type(finn_variable2), intent(out) :: destination2

    if (allocated(source2%ffire)) then
        call initialize_finn_variable2(destination2,
         &
            dim1=size(source2%ffire(:,1,1,1)), &
            dim2=size(source2%ffire(1,:,1,1)), &
            dim3=size(source2%ffire(1,1,:,1)), &
            dim4=size(source2%ffire(1,1,1,:)), &
            firevar_name=source2%firevar_name, &
            units2=source2%units2)
        destination2%ffire = source2%ffire
    else
        call initialize_finn_variable2(destination2, firevar_name=source2%firevar_name, &
         &
            units2=source2%units2)
    endif

end subroutine

subroutine initialize_finn_variable2(finn_var2, dim1, dim2, dim3, dim4, &
    firevar_name, units2)
    implicit none

    type(finn_variable2), intent(out) :: finn_var2
    integer, intent(in), optional :: dim1, dim2, dim3, dim4 !AD plumerise dim4
    character(len=1), intent(in), optional :: firevar_name, units2

    integer :: dummy2, l_dim1, l_dim2, l_dim3, l_dim4
    l_dim1 = 0
    l_dim2 = 0
    l_dim3 = 0
    l_dim4 = 0

    if (present(dim1)) l_dim1 = dim1
    if (present(dim1)) l_dim2 = dim2
    if (present(dim1)) l_dim3 = dim3
    if (present(dim1)) l_dim4 = dim4

    dummy2 = initializeString(finn_var2%firevar_name)
    dummy2 = initializeString(finn_var2%units2)
    if (allocated(finn_var2%ffire)) deallocate(finn_var2%ffire)

    if((present(dim1)) .and. (present(dim2)) .and. (present(dim3)) &
    .and. (present(dim4))) then
        if (.not. allocate_real_4_4D(finn_var2%ffire, l_dim4, l_dim1, l_dim2,
         &
         l_dim3, 'finn_var2%ffire')) stop
    endif

    if (present(firevar_name)) then
        !print *, 'copying species_name "',
      !species_name(1:len(trim(species_name))), ","
        dummy2 = copyString(firevar_name(1:len(trim(firevar_name))), &
         &
         finn_var2%firevar_name)
    endif

    if (present(units2)) then
        !print *, 'copying units "', units(1:len(trim(units))), ","
        dummy2 = copyString(units2(1:len(trim(units2))), finn_var2%units2)
    endif

    return
end subroutine initialize_finn_variable2

# Line 426: Edit read_finn_file function
    type(finn_variable2) :: temp_finn_variable2

# Line 466: Include new finn structural component
    finn_struct%firetype = 0

# Line 511: add new dimensional case
case ('FIR')
    finn_struct%firetype = dimension_length

# Line 578, 590-597: allocations/deallocations
if (allocated(finn_struct%ffvar)) deallocate(finn_struct%ffvar)
if (allocated(finn_struct%ffvar)) deallocate(finn_struct%ffvar)
print *, 'allocating FIRESIZE and MEANFCT'
allocate(finn_struct%ffvar(2), stat=allo_status)
if (allo_status .ne. 0) then
    print *, 'Error (read_finn_file): Unable to allocate storage for ', &
    'FIRESIZE and MEANFCT. Exiting...'  
    stop
endif

#Line 620-651: The special cases of MEANFCT and FIRESIZE
case ('MEA') !MEANFCT
if (read_type .eq. 0) then
    !print *, 'initialize step 0'
    call initialize_finn_variable2(temp_finn_variable2, &
        firevar_name=variable_name, units2='?')
else
    call initialize_finn_variable2(temp_finn_variable2, &
        dim4=finn_struct%firetype, &
        dim1=finn_struct%num_columns, &
        dim2=finn_struct%num_rows, &
        dim3=finn_struct%num_hours, &
        firevar_name=variable_name, &
        units2='?')
endif
    call check(nf90_get_att(ncid, iter, "units",temp_finn_variable2%units2))
call copy_finn_variable2(temp_finn_variable2,finn_struct%ffvar(1))

    if (read_type .eq. 0) then
        !print *, 'initialize step 0'
        call initialize_finn_variable2(temp_finn_variable2, &
            firevar_name=variable_name, units2='?')
    else
        call initialize_finn_variable2(temp_finn_variable2, &
            dim4=finn_struct%firetype, &
            dim1=finn_struct%num_columns, &
            dim2=finn_struct%num_rows, &
            dim3=finn_struct%num_hours, &
            firevar_name=variable_name, &
            units2='?')
    endif
    call check(nf90_get_att(ncid, iter, "units",temp_finn_variable2%units2))
call copy_finn_variable2(temp_finn_variable2,finn_struct%ffvar(2))

#Line 936: New allocate function
logical function allocate_real_4_4D(var, num4, num1, num2, num3, varname)
implicit none
character(len=*) :: varname
real*4, intent(inout), allocatable :: var(:,:,:,:)
integer, intent(inout) :: num1, num2, num3, num4
integer :: allo_status
allocate_real_4_4D = .false.
if (num4 .le. 0 .or. num2 .le. 0 .or. num3 .le. 0 .or. num4 .le. 0) then
    print *, 'Error (allocate_real_4_4D()): Invalid # of records to ', &
    'allocate space for (', num4, ',', num1, ',', num2, ',', num3, &
    ').Skipping variable =', &
    varname, '...'
    num1 = 0
    num2 = 0
    num3 = 0
    num4 = 0
else
    if (allocated(var)) deallocate(var)
allocation(var(num4, num1, num2, num3), stat=allo_status)
allocation(var(num1, num2, num3,num4), stat=allo_status)

if (allo_status .ne. 0) then
  print *, 'Error (allocate_real_4_4D()): Unable to allocate ', &
  '(', num4, ',', num1, ',', num2, ',', num3, ') records for the "', varname, &
  '" array.'
else
  allocate_real_4_4D = .true.
endif
endif
return
end function allocate_real_4_4D

B. add uuwind, vvwind to variable list

integer function read_particle_netcdf(filename, num_times, num_particles, num_levels, &
jjet, ddt, particle_indices, ipos, jpos, kpos, xlon, ylat, wagi, psfc, fcld, &
pvar, qpar, pgc, tgc, tpap, pu0, pv0, pt0, dgc, dns, dagc, qgc, qgcv, smpt, fp, &
ppos, paigs, ptyp, pland, paero, psfcl, prain, vdens, &
vtens, vpre, vprfr, plthg, psfk, kkg, kkt, uuwind, vvwind)
real*4, intent(out), allocatable :: uuwind(:, :, :), vvwind(:, :, :)
case('UUWIND')
if (.not. allocate_real_4_3D(uuwind, num_times, num_particles, num_levels, &
'UUWIND')) stop
  !print *, 'looking for uuwind'
call check(nf90_get_var(ncid, variable_id, uuwind))
case('VVWIND')
if (.not. allocate_real_4_3D(vvwind, num_times, num_particles, num_levels, &
'VVWIND')) stop
  !print *, 'looking for vvwind'
call check(nf90_get_var(ncid, variable_id, vvwind))

• Makefile: add plumerise modules

SRC := $(SRC) read_species_mapping_file.f90 read_particle_netcdf.f90
write_conc_out_csv.f90 module_model_constants.f90 module_zero_plumegen_coms.f90
module_chem_plumerise_scalar.f90

module_model_constants.o: module_model_constants.f90
  $(COMPILER) $(PREPROCESS_FLAG) -c $(COMPILE_FLAGS)
  $(STILT_CHEM_INCLUDES_STATEMENT) $(NETCDF_INCLUDE_STATEMENT) $(MODULES_OUTPUT_CMD)
module_model_constants.f90

module_zero_plumegen_coms.o: module_zero_plumegen_coms.f90
  $(COMPILER) $(PREPROCESS_FLAG) -c $(COMPILE_FLAGS)
  $(STILT_CHEM_INCLUDES_STATEMENT) $(NETCDF_INCLUDE_STATEMENT) $(MODULES_OUTPUT_CMD)
module_zero_plumegen_coms.f90

module_chem_plumerise_scalar.o: module_chem_plumerise_scalar.f90
  $(COMPILER) $(PREPROCESS_FLAG) -c $(COMPILE_FLAGS)
  $(STILT_CHEM_INCLUDES_STATEMENT) $(NETCDF_INCLUDE_STATEMENT) $(MODULES_OUTPUT_CMD)
module_chem_plumerise_scalar.f90

v. FORTRAN 90 PROGRAMS AND SUBROUTINES

# modified: Libraries/hymodelc_backward.F90
# modified: Libraries/hymodelc_forward.F90

• hymodelc_backward.F90 file: allocate, calculate, and write out uuwind, vvwind to PARTICLE.nc file

!ALLOCATE:
real, allocatable :: uuwind(:, :, :)
real, allocatable :: vvwind(:, :, :)
allocate(pzndx(ntstp, numpar), pzpos(ntstp, numpar), &
psigw(ntstp, numpar), pptyp(ntstp, numpar), pland(ntstp, numpar), &
paar(ntstp, numpar), psfcl(ntstp, numpar), prain(ntstp, numpar), &
vden(nstps, numpar, nlvl), vtemp(nstps, numpar, nlvl), &
vpre(nstps, numpar, nlvl), vprfr(nstps, numpar, nlvl), qgc(nstps, numpar, nlvl), &
hymodelc_forward.F90 file: allocate/calculate all necessary input variables needed to run 1-D plumerise model for fires. Run plumerise model and get appropriate species emissions in each layer within the injection height window estimated by the plumerise model. If the plumerise model calculates a maximum injection height of less than (or equal to) 2000m, keep the original set up of evenly distributing emissions within the PBL.

!CALL THE PLUMERISE MODULE(s), allocate necessary variables:
use module_chem_plumerise_scalar
real, allocatable   :: uuwind(:, :, :)
real, allocatable   :: vvwind(:, :, :)
real :: temp_fire_size, temp_mean_fct
integer :: latitude_id, longitude_id, altitude_id
integer :: nonfire_ncid, iter1, co_index, co2_index, finn_ncid
logical :: has_fire_data
integer :: fire_data_status, extra_emissions_Data_status, iaero
integer, allocatable :: fire_vector(:,), nonfire_vector(:,), test_fire_vector(:,)
integer, allocatable :: fire_vector_t1(:,), fire_vector_t2(:,), fire_vector_t3(:,),
& fire_vector_t4(:,)
integer :: imja
integer, parameter :: m1 = 28
integer, parameter :: nspecies = 2
integer :: kplume,nztiter
integer :: narr_ipos
integer :: narr_jpos
integer :: narr_kpos
!eburn_out is in units of kg/m$^3$ and is related to eburn_in (kg/m$^2$) by multiplying with mean_fct (unitless) and 1/dz_flam (thickness of the relevant vertical layer in m)
real,dimension(28,2) :: eburn_out !29 = NARR_heights, 28=NARR_box_heights
real,dimension(2) :: eburn_in
real,dimension(28) :: theta1D,rv1D,pp1D
real,    dimension(28,1,1) :: up, vp, wp,theta,pp,dn0,rv
real,    dimension(28)     :: zt_rams_sum,zt_rams,zm_rams,eburn_frac
real :: eburn_use
integer, parameter :: nztrams = 28
integer, parameter :: nztiter_s = 1
integer, parameter :: nveg_agreg = 4
integer, parameter :: tropical_forest = 1
integer, parameter :: boreal_forest = 2
integer, parameter :: savannah = 3
integer, parameter :: grassland = 4
real, dimension(nveg_agreg) :: firesize,mean_fct
real :: cpplrise,rdplrise
real, dimension(28) :: es,T_c,pvap
real:: plumetop,plumebot
integer, allocatable :: dimdum1,dimdum2

Flags passed from WRF, needed by module_model_constants.f90.

integer, parameter :: NMM_CORE = 0
integer, parameter :: HWRF = 0
allocate(fire_vector_t1(4),fire_vector_t2(4),fire_vector_t3(4),fire_vector_t4(4),
stat=allo_status)
if (allo_status .ne. 0) then
  print *, 'Error (STILT-ASP): Unable to allocate fire type (4D) vectors!
  stop
endif
!
!Add in uuwind, vvwind to the call to the netcdf read subroutine
if (read_particle_netcdf('PARTICLE.nc', num_times, num_particles, num_levels, &
  jjet, ddt, particle_indices, ipos, jpos, xlon, ylat, &
  psfc, fcl, nps, kpos, ppar, xlat, &
  tpar, pu0, pv0, p0, dgc, dns, dsgc, ggc, qgc, &
  smpt, fp, pspos, &
  psigw, ppgw, pland, paero, &
  psfcl, psf, phdwpx, prain, vdens, vtemp, &
  vwind, vpres, &
  vrhfr, plume, plumebot, &
  kkg, kkt, uuwind, vvwind) .lt. 0) then
  print *, 'Error (STILT-ASP): Cannot read in conditions from PARTICLE.nc file!
  stop
! If particle is above mixing layer height, ignore all fire emissions
! AD (AER Inc) FOR plumerise ALWAYS SET count_fire to TRUE
! count_fire = .false.
!if (current_altitude .le. current_pbl_height) then
  count_fire = .true.
!endif
!****************************************************
!AD (AER Inc) Include wrf-chem plumerise call 3/19/19
!*******************************************************************
fire_vector_t1 = (/ipos(is,jp), jpos(is,jp), &
closest_time_index_particle,1 /)
fire_vector_t2 = (/ipos(is,jp), jpos(is,jp), &
closest_time_index_particle,2 /)
fire_vector_t3 = (/ipos(is,jp), jpos(is,jp), &
closest_time_index_particle,3 /)
fire_vector_t4 = (/ipos(is,jp), jpos(is,jp), &
closest_time_index_particle,4 /)
MJA Read in FIRESIZE for each of the 4 types
  call check(nf90_inq_varid(finn_ncid, 'FIRESIZE',latitude_var_id))
call check(nf90_get_var(finn_ncid, latitude_var_id, &
  temp_fire_size, start=fire_vector_t1))
firesize(1) = temp_fire_size
call check(nf90_get_var(finn_ncid, latitude_var_id, &
  temp_fire_size, start=fire_vector_t2))
firesize(2) = temp_fire_size
call check(nf90_get_var(finn_ncid, latitude_var_id, &
  temp_fire_size, start=fire_vector_t3))
firesize(3) = temp_fire_size
call check(nf90_get_var(finn_ncid, latitude_var_id, &
  temp_fire_size, start=fire_vector_t4))
firesize(4) = temp_fire_size
if (sum(firesize) .gt. 0.) then !CHECK IF FIRES: ONLY IF FIRES EXIST!
call check(nf90_inq_varid(finn_ncid, 'MEANFCT',latitude_var_id))
call check(nf90_get_var(finn_ncid, latitude_var_id, &
  temp_mean_fct, start=fire_vector_t1))
  mean_fct(1) = temp_mean_fct
call check(nf90_get_var(finn_ncid, latitude_var_id, &
  temp_mean_fct, start=fire_vector_t2))
  mean_fct(2) = temp_mean_fct
call check(nf90_get_var(finn_ncid, latitude_var_id, &
  temp_mean_fct, start=fire_vector_t3))
  mean_fct(3) = temp_mean_fct

call check(nf90_get_var(finn_ncid, latitude_var_id, &
  temp_mean_fct, start=fire_vector_t4))

mean_fct(4) = temp_mean_fct

eburn_in = (/1.1E7, 2.3E6/) !In molecules. dummy for now

mean_fct = (/1.0, 0.0, 0.0, 0.0, 0.0/) !Fraction of each firetype.

up = reshape(uwind(is,jp,1:28),shape(up)) !u-wind profile

vp = reshape(vwind(is,jp,1:28),shape(vp)) !v-wind profile

theta1D= vtemp(is,jp,1:28)*1000/vpres(is,jp,1:28)**0.288

theta = reshape(theta1D, shape(theta)) !potential temp

rdplrise = 287 !ideal gas constant

cpplrise = 7.*rdplrise/2. !specific heat constant pressure

pp1D=cpplrise*(vpres(is,jp,1:28)/1000)&**(rdplrise/cpplrise)

pp = reshape(pp1D,shape(pp)) !exner function, form of !
  potential temp

T_c = vtemp(is,jp,1:28)-273.15 !Temp in C

es = 6.1094*exp((17.625*T_c)/(T_c+243.04)) !Sat vap p hPa

pvap = ggcv(is,jp,1:28)*es !Vap press hPa. ggcv = rh/100

rv1D = 621.97*pvap/(vpres(is,jp,1:28)-pvap) !water vap mix ratio g/kg

rv = reshape(rv1D/1000,shape(rv)) !water vap mix ratio as kg/kg

narr_ipos = ipos(is,jp)

narr_jpos = jpos(is,jp)

narr_kpos = kpos(is,jp)

zt_rams = reshape(narr_box_height(narr_ipos,narr_jpos,1:28,imr) &
  ,shape(zt_rams))

!Box heights are heights of individual levels.
!Need to cumulatively sum to get relative to ground lev

zt_rams_sum(nztiter_s) = zt_rams(nztiter_s)

DO nztiter=2,28
    zt_rams_sum(nztiter) = zt_rams_sum(nztiter-1)+&
    zt_rams(nztiter)
ENDDO

zt_rams = zt_rams_sum

zm_rams = reshape(narr_height(narr_ipos,narr_jpos,1:28,imr) &
  ,shape(zm_rams))

!THE ACTUAL PLUMERISE CALL

call plumerise(m1,1,1,1,1,1,1,firesize,mean_fct &
  ,nspecies,eburn_in,eburn_out &
  ,up,vp,wp,theta,pp,dn0,rv,zt_rams,zm_rams)

!Debug: write out eburn_out emissions in each plume layer

write(*,*) mean_fct

do kplume = 1, m1
    write(*,*) kplume, eburn_out(kplume,1)
endo

!Get the top and bottom of injection height window on NARR levels. Determined
!by searching for non-zero instances of emissions in molecules. The first
!layer of the eburn_out is ignored, that is where the emissions are 100% of
!eburn_in.

do kplume = 2, m1
    if (eburn_out(kplume,1) .eq. 0.) then
        dimdum1 = kplume
    endif
    if (eburn_out(kplume+1,1) .ne. 0.) EXIT
end do

plumebot = zm_rams(dimdum1+1)
print *, "PLUMEBOT (m)", plumebot

do kplume = 1, m1
    if (eburn_out(m1-kplume,1) .eq. 0.) then
        dimdum2 = m1-kplume
    endif
    if (eburn_out(m1-kplume,1) .ne. 0.) EXIT
end do
plumetop = zm_rams(dimdum2-1)
print *, "PLUMETOP (m)", plumetop
eburn_frac() = 0.0
do imja = 2, 28
  eburn_frac(imja) = eburn_out(imja,1)/eburn_in(1) !units are m
end do

 eburn_use = eburn_frac(narr_kpos)
else !IF NO FIRES FROM FINN FILE
  eburn_use = 1.0
  plumetop = current_pbl_height
  plumebot = current_pbl_height
endif !END CHECK IF FIRES
!eburn_frac is in units of 1/m And this is a value for each grid box level
!which is then used in place of current_pbl_height This ratio will be the same for
!all species so we just do this once. eburn_in is qsmold in kg/m2
!eburn_out is mean_fct(unitless)*qsmold(kg/m2)*dzi(m) = kg/m3
!(note that dzi = 1/dz_flam where dz_flam is thickness of the relevant vertical
!layer. Related to zzcon(k2)-zzcon(k1) which in turn is derived from zm_rams.

!eburn_frac(narr_kpos) gives the appropriate level for the relevant particle.
!In later parts of the fire code we then replace all instances of /current_pbl_height
!with *eburn_frac(narr_kpos) to account for the emissions in the
!vertical layer associated with a plume top height > 2000m
! (or whatever current_pbl_height is set at).
!If plumetop is.le. current_pbl_height, then the original method is used.

!Sample output of eburn_out and plumetop, plumebot calc:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>11000000.0</td>
</tr>
<tr>
<td>2</td>
<td>0.00000000</td>
</tr>
<tr>
<td>3</td>
<td>9087.32812</td>
</tr>
<tr>
<td>4</td>
<td>9087.32812</td>
</tr>
<tr>
<td>5</td>
<td>9087.32812</td>
</tr>
<tr>
<td>6</td>
<td>9087.32812</td>
</tr>
<tr>
<td>7</td>
<td>9087.32812</td>
</tr>
<tr>
<td>8</td>
<td>0.00000000</td>
</tr>
<tr>
<td>9</td>
<td>0.00000000</td>
</tr>
<tr>
<td>10</td>
<td>0.00000000</td>
</tr>
<tr>
<td>11</td>
<td>0.00000000</td>
</tr>
<tr>
<td>12</td>
<td>0.00000000</td>
</tr>
<tr>
<td>13</td>
<td>0.00000000</td>
</tr>
<tr>
<td>14</td>
<td>0.00000000</td>
</tr>
<tr>
<td>15</td>
<td>0.00000000</td>
</tr>
<tr>
<td>16</td>
<td>0.00000000</td>
</tr>
<tr>
<td>17</td>
<td>0.00000000</td>
</tr>
<tr>
<td>18</td>
<td>0.00000000</td>
</tr>
<tr>
<td>19</td>
<td>0.00000000</td>
</tr>
<tr>
<td>20</td>
<td>0.00000000</td>
</tr>
<tr>
<td>21</td>
<td>0.00000000</td>
</tr>
<tr>
<td>22</td>
<td>0.00000000</td>
</tr>
<tr>
<td>23</td>
<td>0.00000000</td>
</tr>
<tr>
<td>24</td>
<td>0.00000000</td>
</tr>
<tr>
<td>25</td>
<td>0.00000000</td>
</tr>
<tr>
<td>26</td>
<td>0.00000000</td>
</tr>
<tr>
<td>27</td>
<td>0.00000000</td>
</tr>
<tr>
<td>28</td>
<td>0.00000000</td>
</tr>
</tbody>
</table>

PLUMEBOT (m)  524
PLUMETOP (m) 1502

!********************************************************************
! END wrf-chem PLUMERI CALL
!********************************************************************

!Now for each of nspecies, distribute the emissions evenly in the appropriate layer as
!determined by the plumerise call
if (plumetop .le. current_pbl_height) then
    if (current_altitude .le. current_pbl_height) then
        delta_conc_fire_emissions_aerosols_numconc(:) =
            delta_conc_fire_emissions_aerosols_numconc(:)*abs(ddt(is))/60.0/
                current_pbl_height
    else
        delta_conc_fire_emissions_aerosols_numconc(:)=0.0
    endif
    !particles/m3
else
    !particles/m3 distributed in the narr layer
    !corresponding to the layer the stilt
    !particle is in (eburn_frac units are 1/m)
    delta_conc_fire_emissions_aerosols_numconc(:) =
        delta_conc_fire_emissions_aerosols_numconc(:)*abs(ddt(is))/60.0*
            eburn_frac(narr_kpos)
endif

if (plumetop .le. current_pbl_height) then
    if (current_altitude .le. current_pbl_height) then
        delta_conc_fire_emissions_aerosols_numconc(1) = temp_finn_conc*(
            abs(ddt(is))/60.0)/current_pbl_height/1.0e6 !convert to particles/cm3
    else
        delta_conc_fire_emissions_aerosols_numconc(1)=0.0
    endif
else
    delta_conc_fire_emissions_aerosols_numconc(1)= temp_finn_conc*(
        abs(ddt(is))/60.0)*eburn_use/1.0e6 !convert to particles/cm3
endif

if (plumetop .le. current_pbl_height) then
    if (current_altitude .le. current_pbl_height) then
        if (fire_variable_name(1:5) .eq. 'IVOC1') then
            ivoc1_conc = ivoc1_conc + temp_finn_conc*(abs(ddt(&
                is))/60.0)*mw_temp/current_pbl_height
        else
            bc_conc = bc_conc + temp_finn_conc*(abs(ddt(is))/60.0)*mw_temp/
                current_pbl_height
        endif
    else if (current_altitude .gt. current_pbl_height) then
        if (fire_variable_name(1:5) .eq. 'IVOC1') then
            ivoc1_conc = ivoc1_conc
        else
            bc_conc = bc_conc
        endif
    endif
else
    if (fire_variable_name(1:5) .eq. 'IVOC1') then
        ivoc1_conc = ivoc1_conc +temp_finn_conc*(abs(ddt(&
            is))/60.0)*mw_temp*eburn_use
    else
        bc_conc = bc_conc +temp_finn_conc*(abs(ddt(is))/60.0)*mw_temp*&
            eburn_use
    endif
endif

if (plumetop .le. current_pbl_height) then
    if (current_altitude .le. current_pbl_height) then
        delta_conc_fire_emissions_aerosols_massconc(1,
            org_phase_chemical_index + &
            HowManyAqChems + HowManyAqCations + &
            HowManyAqAnions) = temp_finn_conc*(abs(ddt(&
                is))/60.0)*mw_temp/current_pbl_height
    else
        delta_conc_fire_emissions_aerosols_massconc(1,
            org_phase_chemical_index+ &
            HowManyAqChems + HowManyAqCations + &
            HowManyAqAnions) = 0
    endif
else
    delta_conc_fire_emissions_aerosols_massconc(1,
        org_phase_chemical_index + &
        HowManyAqChems + HowManyAqCations + &
        HowManyAqAnions) = temp_finn_conc*(abs(ddt(is))/60.0)*mw_temp/
            current_pbl_height
    endif
if (plumetop .le. current_pbl_height) then
  if (current_altitude .le. current_pbl_height) then
    delta_conc_fire_emissions_aerosols_massconc(1,aq_phase_chemical_index) = &
    temp_finn_conc*(abs(ddt(is))/60.0)*mw_temp/current_pbl_height
  else
    delta_conc_fire_emissions_aerosols_massconc(1,aq_phase_chemical_index) = 0
  endif
else
  delta_conc_fire_emissions_aerosols_massconc(1,aq_phase_chemical_index) = &
  temp_finn_conc &
  *(abs(ddt(is))/60.0)*mw_temp*eburn_use !Convert to ug/m3
endif

if (plumetop .le. current_pbl_height) then
  if (current_altitude .le. current_pbl_height) then
    kcl_conc = kcl_conc + temp_finn_conc*(abs(ddt(is))/&
    60.0)*mw_temp/current_pbl_height
  else
    kcl_conc = kcl_conc
  endif
else
  kcl_conc = kcl_conc + temp_finn_conc*(abs(ddt(is))/&
  60.0)*mw_temp*eburn_use
endif

if (plumetop .le. current_pbl_height) then
  if (current_altitude .le. current_pbl_height) then
    delta_conc_fire_emissions_gases(gas_phase_chemical_index) = &
    temp_finn_conc*(abs(ddt(is))/60.0)*(tpar(is,jp)*8.314/(&
    100.0*ppar(is,jp)))*current_pbl_height
  else
    delta_conc_fire_emissions_gases(gas_phase_chemical_index) = 0
  endif
else
  delta_conc_fire_emissions_gases(gas_phase_chemical_index) = &
  temp_finn_conc*(abs(ddt(is))/60.0)*(tpar(is,jp)*8.314/(&
  100.0*ppar(is,jp)))*eburn_use
endif