Novel Database and Usage Analytics for CESM Climate Model

First Steps to Tracking Worldwide Configuration and Performance

Lolita Mannik

Regis University

National Center for Atmospheric Research (NCAR)

Summer Internships in Parallel Computational Science (SIParCS)

This material is based upon work supported by the National Center for Atmospheric Research, which is a major facility sponsor ed by the National Science Foundation under Cooperative Agreement No. 1852977.

Overview

1. Background on Community Earth System Model (CESM)
2. Model's configuration
3. Data preparation and analysis
4. Key findings
5. Conclusion and future work

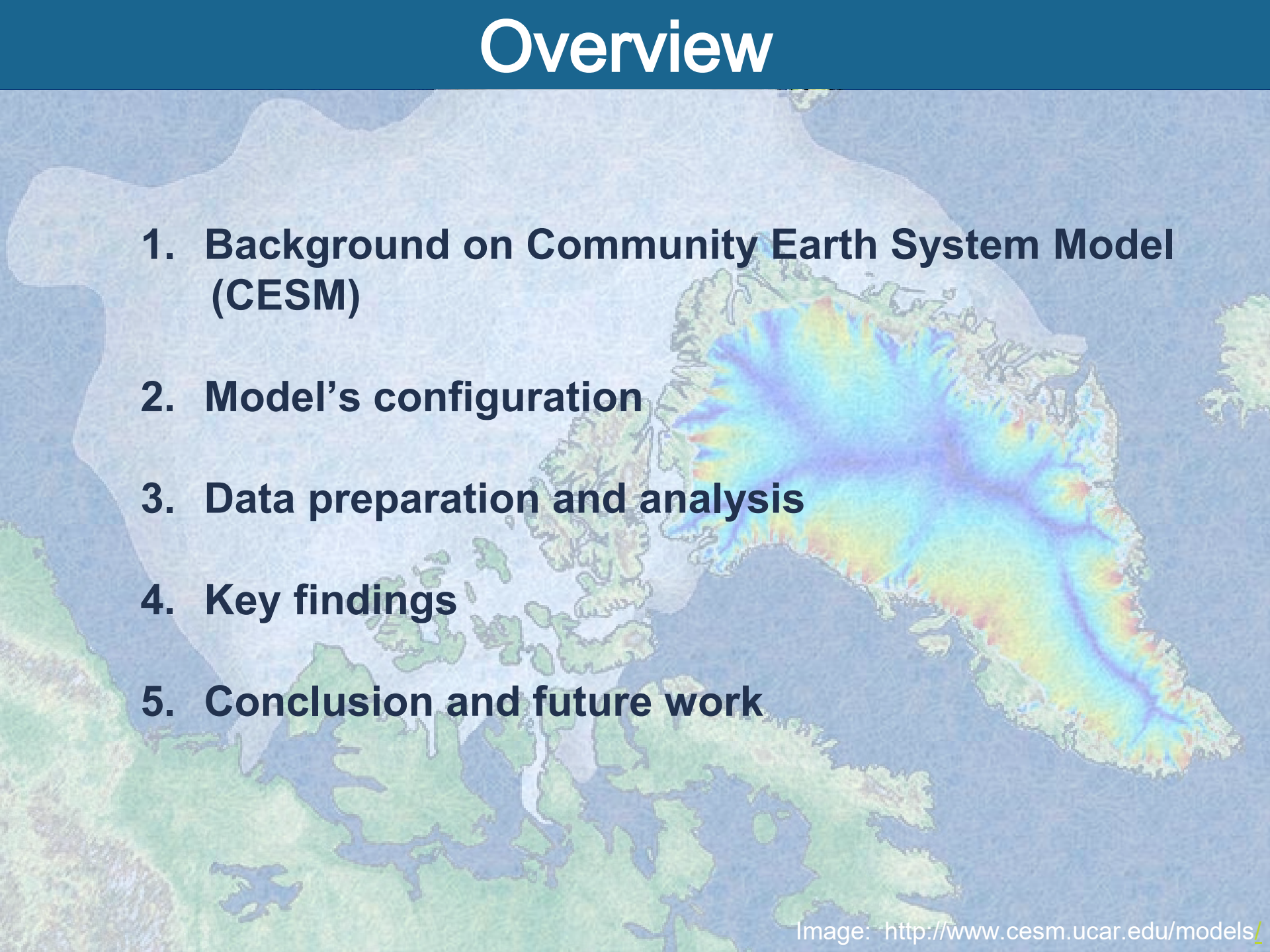


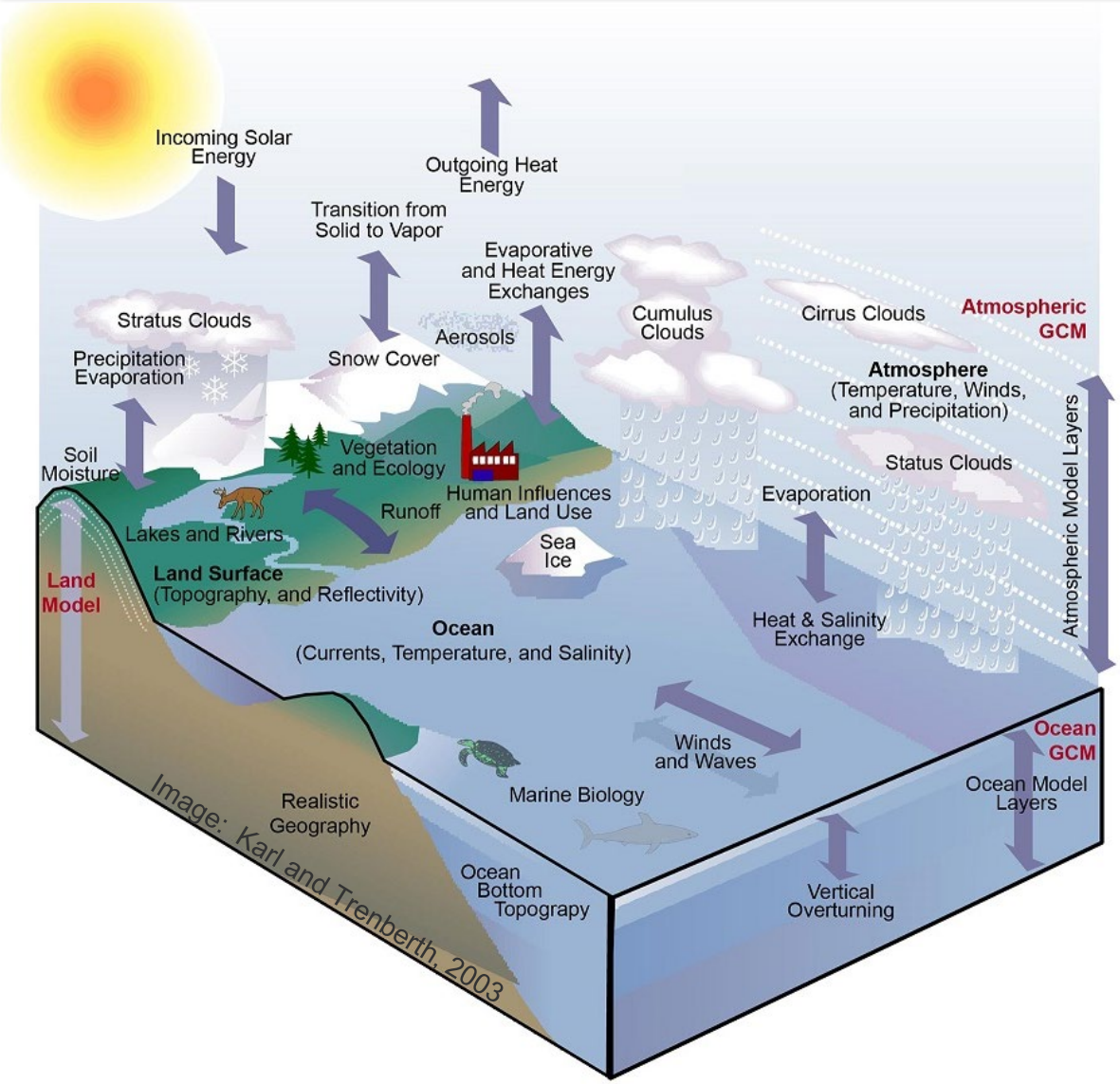
Image: http://www.cesm.ucar.edu/models/

Goal

Create a database dedicated to tracking broader CESM usage and performance

- Learn how the scientists use the model
- Track computational performance

CESM Climate Model



- Virtual laboratory
- Freely available
- Components:
  - Atmosphere
  - Land
  - Ocean
  - River
  - Sea and Land Ice
  - Wave

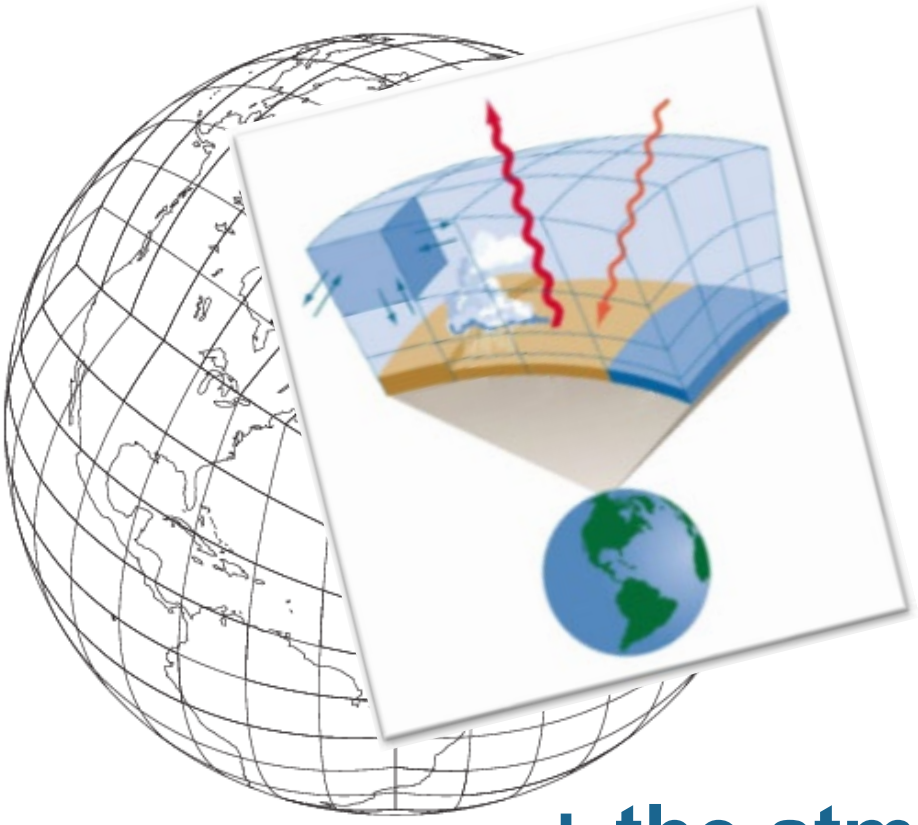
CESM = Community Earth System Model

Resolution

Coarse

Fine

Resolution

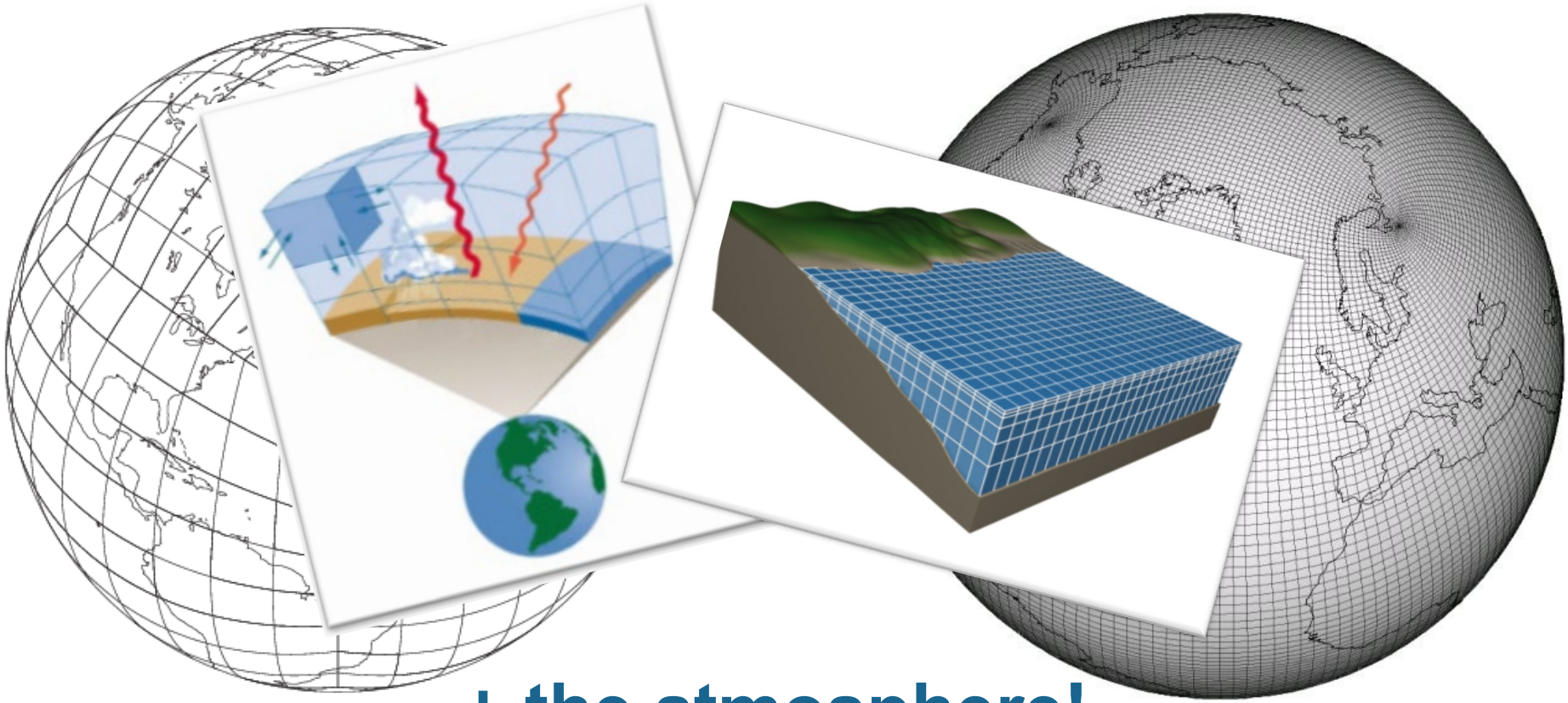


+ the atmosphere!

Coarse **Fine**

Images: K. Cantner, AGI; ETH, Zurich – Climate Model (Ruddiman, 2001),
http://stream1.cmatc.cn/pub/comet/MarineMeteorologyOceans/ocean_models/comet/oceans/ocean_models/print.htm

Resolution



+ the atmosphere!

+ the ocean!

Coarse **Fine**

Images: K. Cantner, AGI; ETH, Zurich – Climate Model (Ruddiman, 2001),
http://stream1.cmatc.cn/pub/comet/MarineMeteorologyOceans/ocean_models/comet/oceans/ocean_models/print.htm

Method

1

Data Engineering

- Acquiring
- Saving

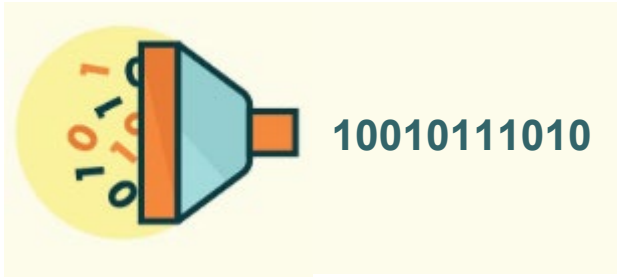
```
---------------- TIMING PROFILE ---------------------
   Case        : b.e21.BHIST.f09_g17.CMIP6-historical.001
   LID         : 2979765.chadmin1.181015-050236

   User        : cmip6
   Curr Date   : Mon Oct 15 10:01:22 2018
   grid        :
a%0.9x1.25_l%0.9x1.25_oi%gx1v7_r%r05_g%gland4_w%ww3a_m%gx1v7
   compset     : HIST_CAM60_CLM50%BGC-CROP_CICE_POP2%ECO%ABIO-
                 DIC MOSART CISM2%NOEVOLVE WW3 BGC%BDRD
   run_type    : hybrid, continue_run = TRUE (inittype = FALSE)
   stop_option : nyears, stop_n = 5
   run_length  : 1825 days (1825.0 for ocean)

   Init Time   :      63.817 seconds
   Run Time    :   17837.627 seconds        9.774 seconds/day
   Final Time  :       0.057 seconds
```

Method

1

Data Engineering

- Acquiring
- Saving



2

Data Wrangling

- Reindexing
- More parsing
- Set data types
- Intuitive columns
- Calculations

Data Wrangling

Parse Run_Length

3650 days (3650.0 for ocean)

```
#1. Strip everything after "days" in run_length column
df['run_length_temp'] = df['run_length'].str.split('(').str[0]

#Confirm every run_length contains the same units of days
substr = 'days'
print ("Rows in df:", len(df))
print ("Rows with units of days:", df.run_length_temp.str.count(substr).sum())
```

```
Rows in df: 5160
Rows with units of days: 5160
```

```
#2. Strip "days" in run_length column
df['run_length_days'] = df['run_length_temp'].str.split('d').str[0]
df.run_length_days.unique()
```

```
array(['3650 ', '365 ', '730 ', '2 ', '31 ', '1825 ', '2189 ', '1095 ',
       '5840 ', '5475 ', '1460 ', '2190 ', '5 ', '1 ', '10950 ', '7300 ',
       '4014 ', '426 ', '90 ', '4379 '], dtype=object)
```

```
#Convert necessary columns to numeric format
for col in df.columns:
   if 'length_days' in col:
      df[col] = pd.to_numeric(df[col])
```

Data Wrangling

Parse Run_Length

3650 days (3650.0 for ocean)

```
#1. Strip everything after "days" in run_length column
df['run_length_temp'] = df['run_length'].str.split('(').str[0]

#Confirm every run_length contains the same units of days
substr = 'days'
print ("Rows in df:", len(df))
print ("Rows with units of days:", df.run_length_temp.str.count(substr).sum())
```

```
Rows in df: 5160
Rows with units of days: 5160
```

```
#2. Strip "days" in run_length column
df['run_length_days'] = df['run_length_temp'].str.split('d').str[0]
df.run_length_days.unique()
```

```
array(['3650 ', '365 ', '730 ', '2 ', '31 ', '1825 ', '2189 ', '1095 ',
       '5840 ', '5475 ', '1460 ', '2190 ', '5 ', '1 ', '10950 ', '7300 ',
       '4014 ', '426 ', '90 ', '4379 '], dtype=object)
```

```
#Convert necessary columns to numeric format
for col in df.columns:
   if 'length_days' in col:
      df[col] = pd.to_numeric(df[col])
```

Data Wrangling

Parse Run_Length

3650 days (3650.0 for ocean)

```
#1. Strip everything after "days" in run_length column
df['run_length_temp'] = df['run_length'].str.split('(').str[0]

#Confirm every run_length contains the same units of days
substr = 'days'
print ("Rows in df:", len(df))
print ("Rows with units of days:", df.run_length_temp.str.count(substr).sum())
```

```
Rows in df: 5160
Rows with units of days: 5160
```

```
#2. Strip "days" in run_length column
df['run_length_days'] = df['run_length_temp'].str.split('d').str[0]
df.run_length_days.unique()
```

```
array(['3650 ', '365 ', '730 ', '2 ', '31 ', '1825 ', '2189 ', '1095 ',
       '5840 ', '5475 ', '1460 ', '2190 ', '5 ', '1 ', '10950 ', '7300 ',
       '4014 ', '426 ', '90 ', '4379 '], dtype=object)
```

```
#Convert necessary columns to numeric format
for col in df.columns:
   if 'length_days' in col:
      df[col] = pd.to_numeric(df[col])
```

Data Wrangling

Parse Run_Length

3650 days

```
#1. Strip everything after "days" in run_length column
df['run_length_temp'] = df['run_length'].str.split('(').str[0]

#Confirm every run_length contains the same units of days
substr = 'days'
print ("Rows in df:", len(df))
print ("Rows with units of days:", df.run_length_temp.str.count(substr).sum())
```

```
Rows in df: 5160
Rows with units of days: 5160
```

```
#2. Strip "days" in run_length column
df['run_length_days'] = df['run_length_temp'].str.split('d').str[0]
df.run_length_days.unique()
```

```
array(['3650 ', '365 ', '730 ', '2 ', '31 ', '1825 ', '2189 ', '1095 ',
       '5840 ', '5475 ', '1460 ', '2190 ', '5 ', '1 ', '10950 ', '7300 ',
       '4014 ', '426 ', '90 ', '4379 '], dtype=object)
```

```
#Convert necessary columns to numeric format
for col in df.columns:
   if 'length_days' in col:
      df[col] = pd.to_numeric(df[col])
```

Data Wrangling

Parse Run_Length

3650 days

```
#1. Strip everything after "days" in run_length column
df['run_length_temp'] = df['run_length'].str.split('(').str[0]

#Confirm every run_length contains the same units of days
substr = 'days'
print ("Rows in df:", len(df))
print ("Rows with units of days:", df.run_length_temp.str.count(substr).sum())
```

```
Rows in df: 5160
Rows with units of days: 5160
```

```
#2. Strip "days" in run_length column
df['run_length_days'] = df['run_length_temp'].str.split('d').str[0]
df.run_length_days.unique()
```

```
array(['3650 ', '365 ', '730 ', '2 ', '31 ', '1825 ', '2189 ', '1095 ',
       '5840 ', '5475 ', '1460 ', '2190 ', '5 ', '1 ', '10950 ', '7300 ',
       '4014 ', '426 ', '90 ', '4379 '], dtype=object)
```

```
#Convert necessary columns to numeric format
for col in df.columns:
   if 'length_days' in col:
      df[col] = pd.to_numeric(df[col])
```

Data Wrangling

Parse Run_Length

3650

```
#1. Strip everything after "days" in run_length column
df['run_length_temp'] = df['run_length'].str.split('(').str[0]

#Confirm every run_length contains the same units of days
substr = 'days'
print ("Rows in df:", len(df))
print ("Rows with units of days:", df.run_length_temp.str.count(substr).sum())
```

```
Rows in df: 5160
Rows with units of days: 5160
```

```
#2. Strip "days" in run_length column
df['run_length_days'] = df['run_length_temp'].str.split('d').str[0]
df.run_length_days.unique()
```

```
array(['3650 ', '365 ', '730 ', '2 ', '31 ', '1825 ', '2189 ', '1095 ',
       '5840 ', '5475 ', '1460 ', '2190 ', '5 ', '1 ', '10950 ', '7300 ',
       '4014 ', '426 ', '90 ', '4379 '], dtype=object)
```

```
#Convert necessary columns to numeric format
for col in df.columns:
    if 'length_days' in col:
        df[col] = pd.to_numeric(df[col])
```

Data Wrangling

Parse Run_Length

3650

```
#1. Strip everything after "days" in run_length column
df['run_length_temp'] = df['run_length'].str.split('(').str[0]

#Confirm every run_length contains the same units of days
substr = 'days'
print ("Rows in df:", len(df))
print ("Rows with units of days:", df.run_length_temp.str.count(substr).sum())
```

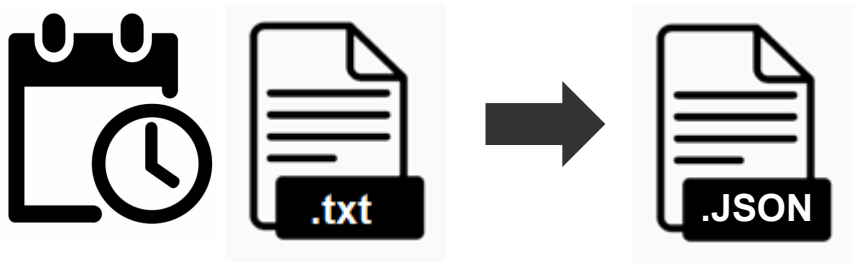
```
Rows in df: 5160
Rows with units of days: 5160
```

```
#2. Strip "days" in run_length column
df['run_length_days'] = df['run_length_temp'].str.split('d').str[0]
df.run_length_days.unique()
```

```
array(['3650 ', '365 ', '730 ', '2 ', '31 ', '1825 ', '2189 ', '1095 ',
       '5840 ', '5475 ', '1460 ', '2190 ', '5 ', '1 ', '10950 ', '7300 ',
       '4014 ', '426 ', '90 ', '4379 '], dtype=object)
```

```
#Convert necessary columns to numeric format
for col in df.columns:
    if 'length_days' in col:
        df[col] = pd.to_numeric(df[col])
```

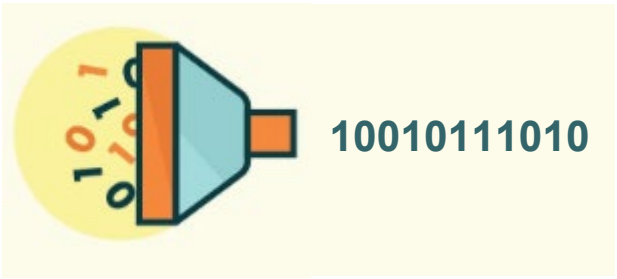
Method



1

Data Engineering

- Acquiring
- Saving



2

Data Wrangling

- Reindexing
- More parsing
- Set data types
- Intuitive columns
- Calculations

3

Data Storage

Method

1

Data Engineering

- Acquiring
-

USABLE DATA!!

ling

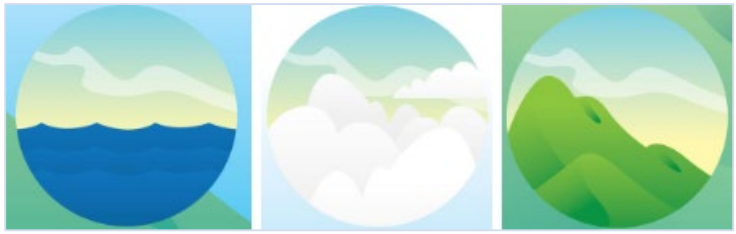
- types
- Intuitive columns
- Calculations

3

Data Storage

Method

4 **Exploratory Data Analysis**



45 unique
component configurations

7 unique grid configurations

Method

4 Exploratory Data Analysis

5 Statistical Analysis

Method

4 Exploratory Data Analysis

5 Statistical Analysis

6 Visualization

Analysis: Yearly Totals

361 Days

106
Unique Experiments

18,469
Simulated Years

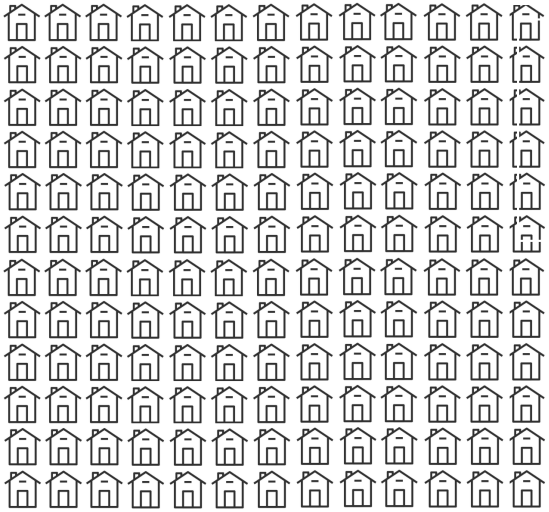
118,824,082
CPU Hours

Power Equivalence

118,824,082
CPU Hours

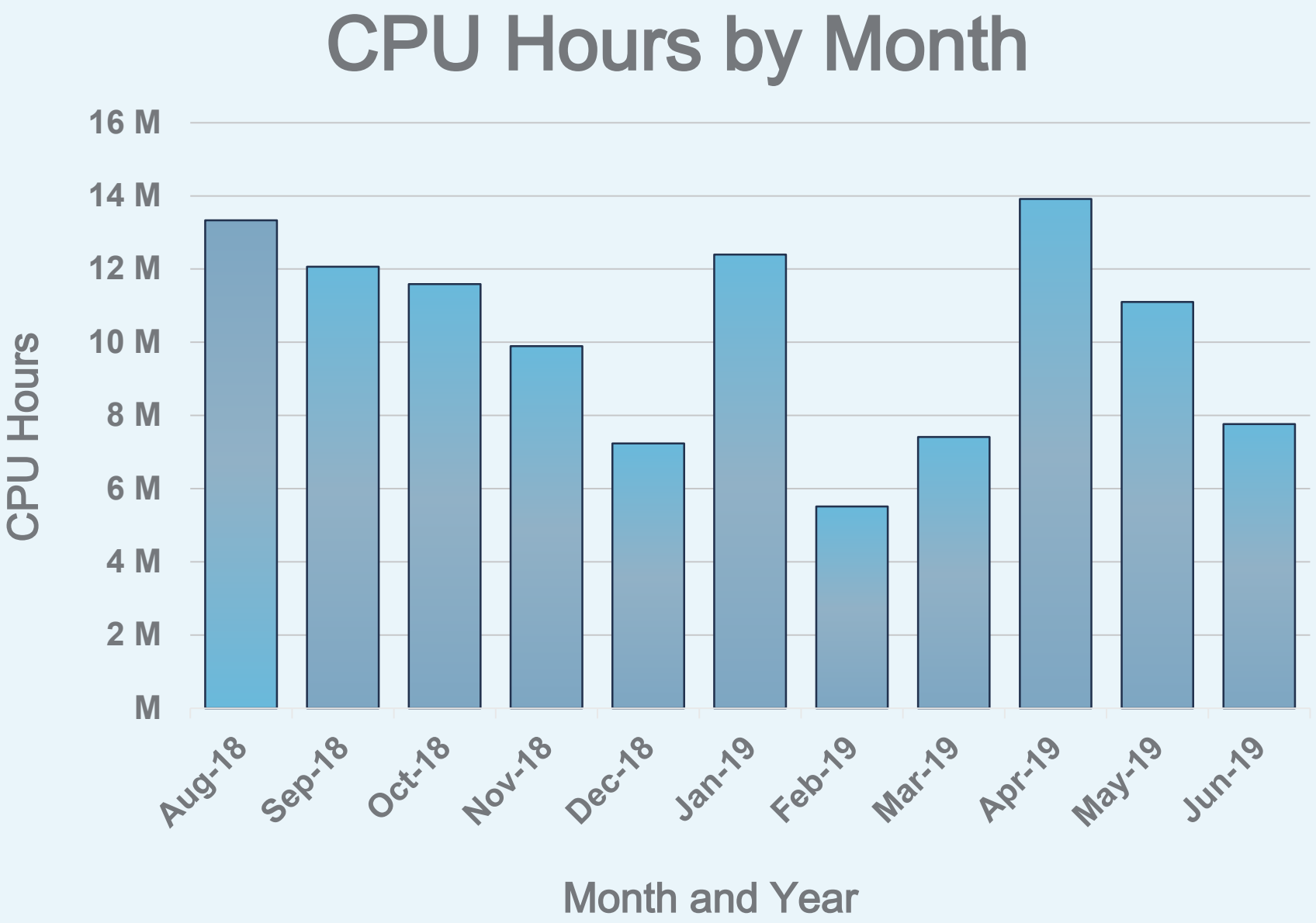
189 trips around the equator in a Nissan Leaf

or

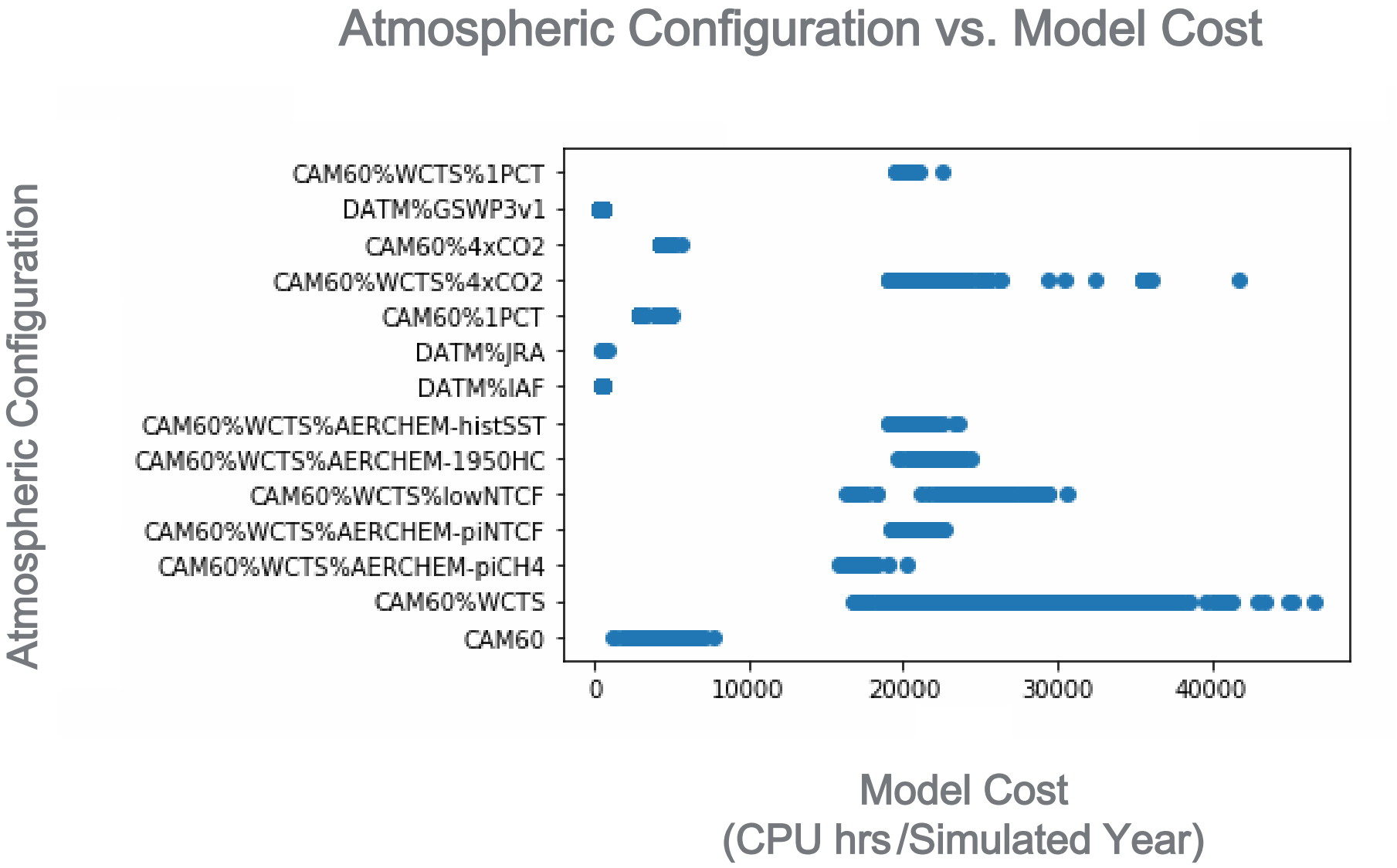


Annual power for 156 Colorado homes

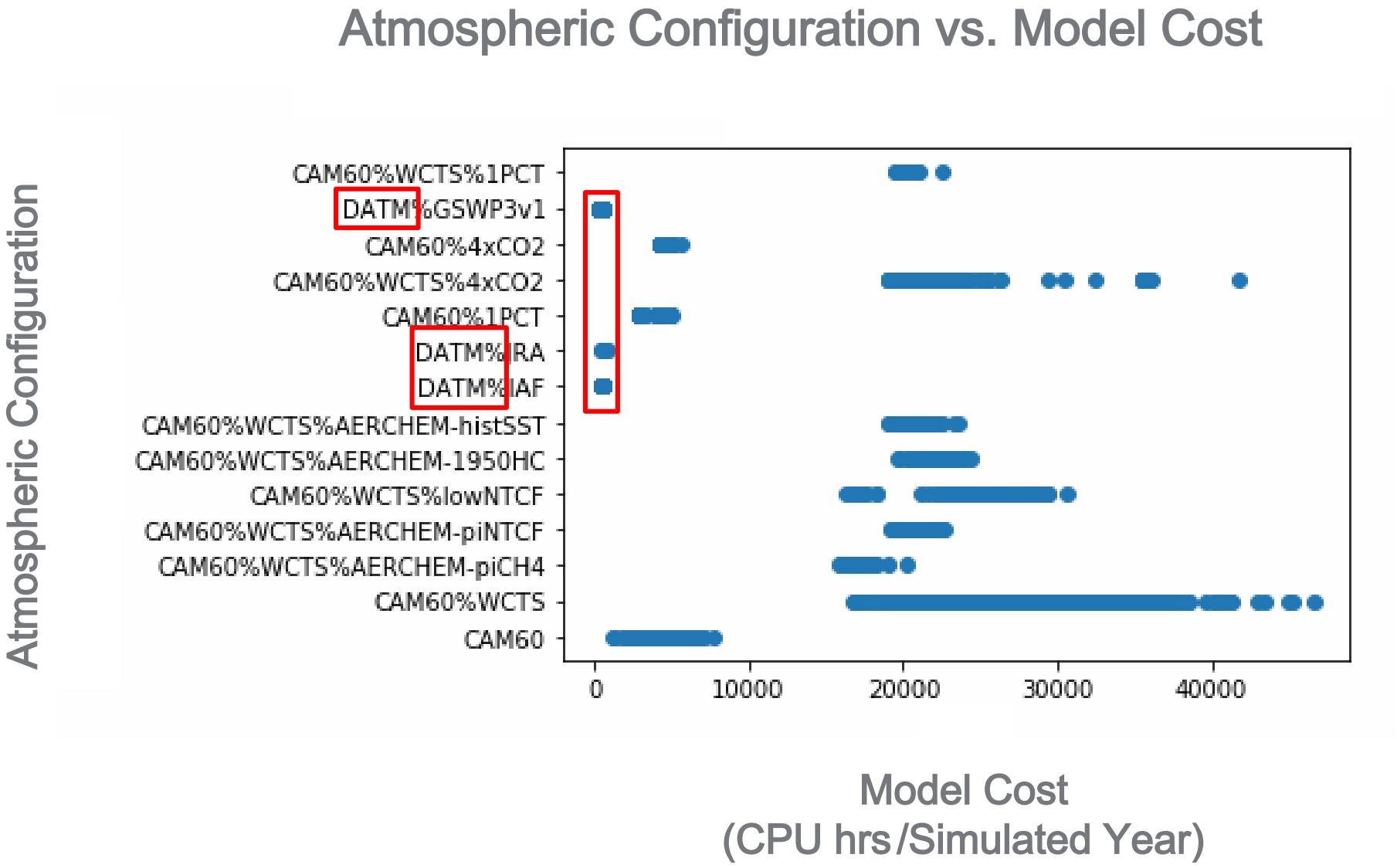
Analysis: Monthly Totals



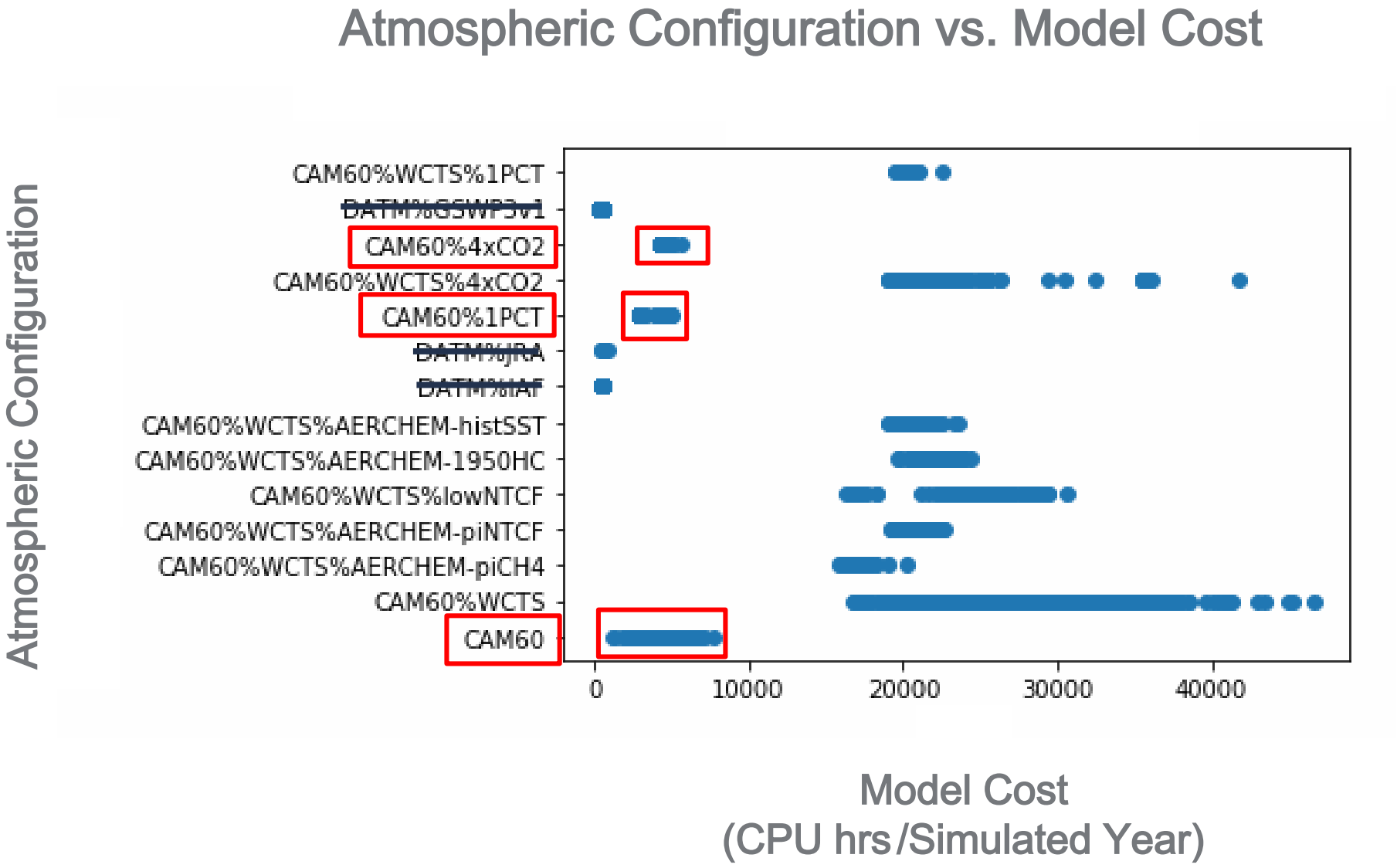
Analysis: Grouping Atmospheric Configurations



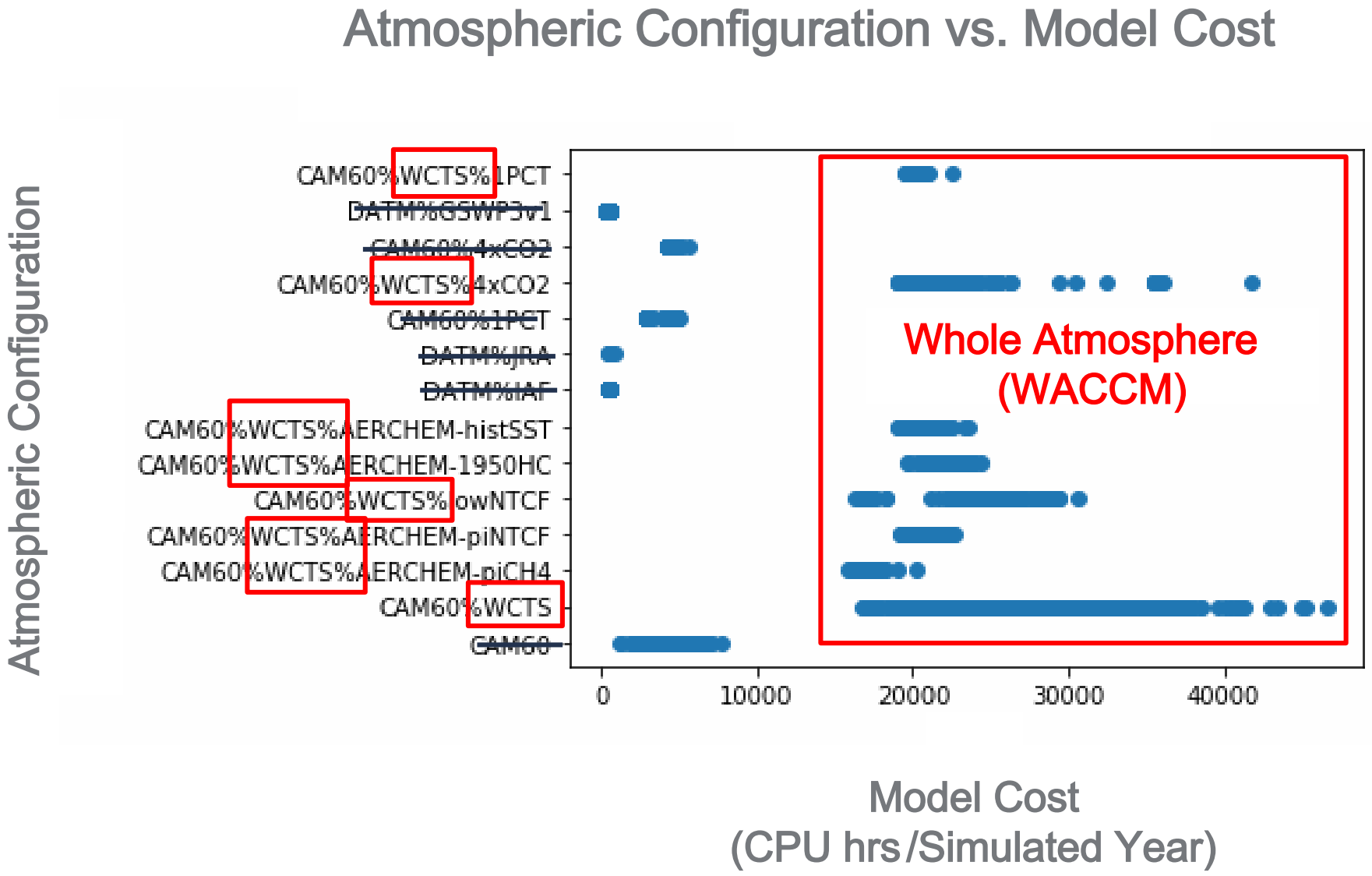
Analysis: Grouping Atmospheric Configurations



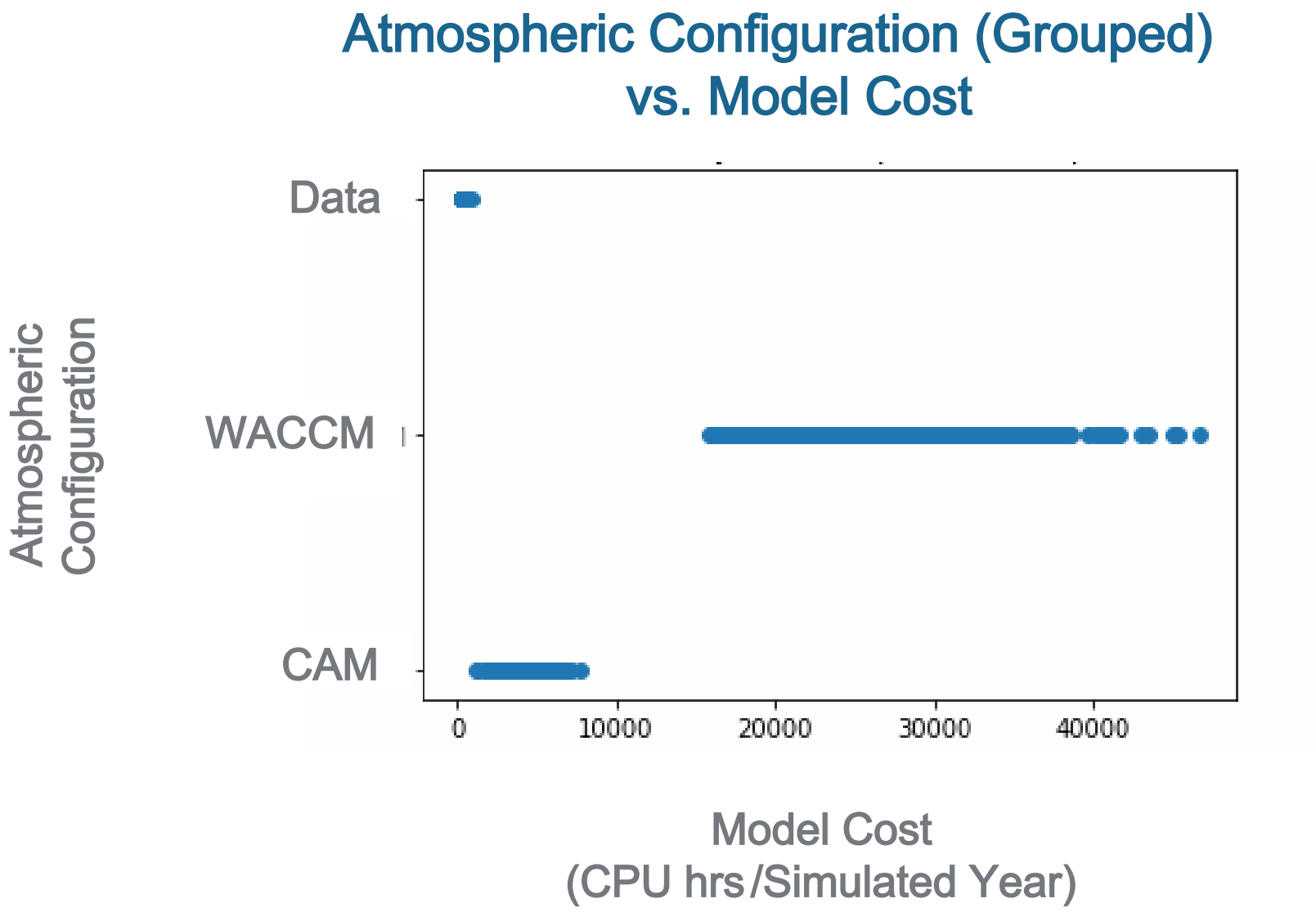
Analysis: Grouping Atmospheric Configurations



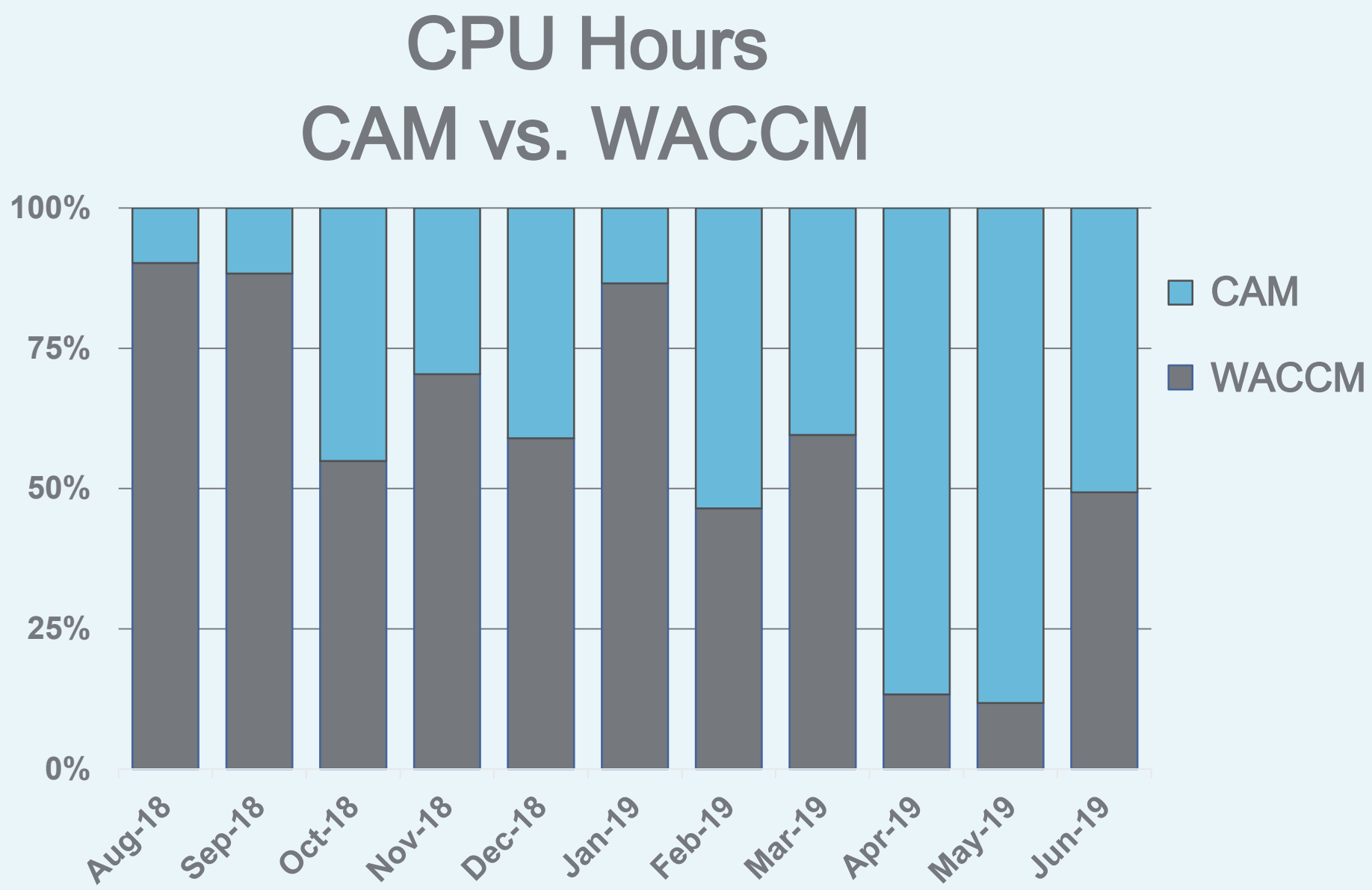
Analysis: Grouping Atmospheric Configurations



Analysis: Grouping Atmospheric Configurations

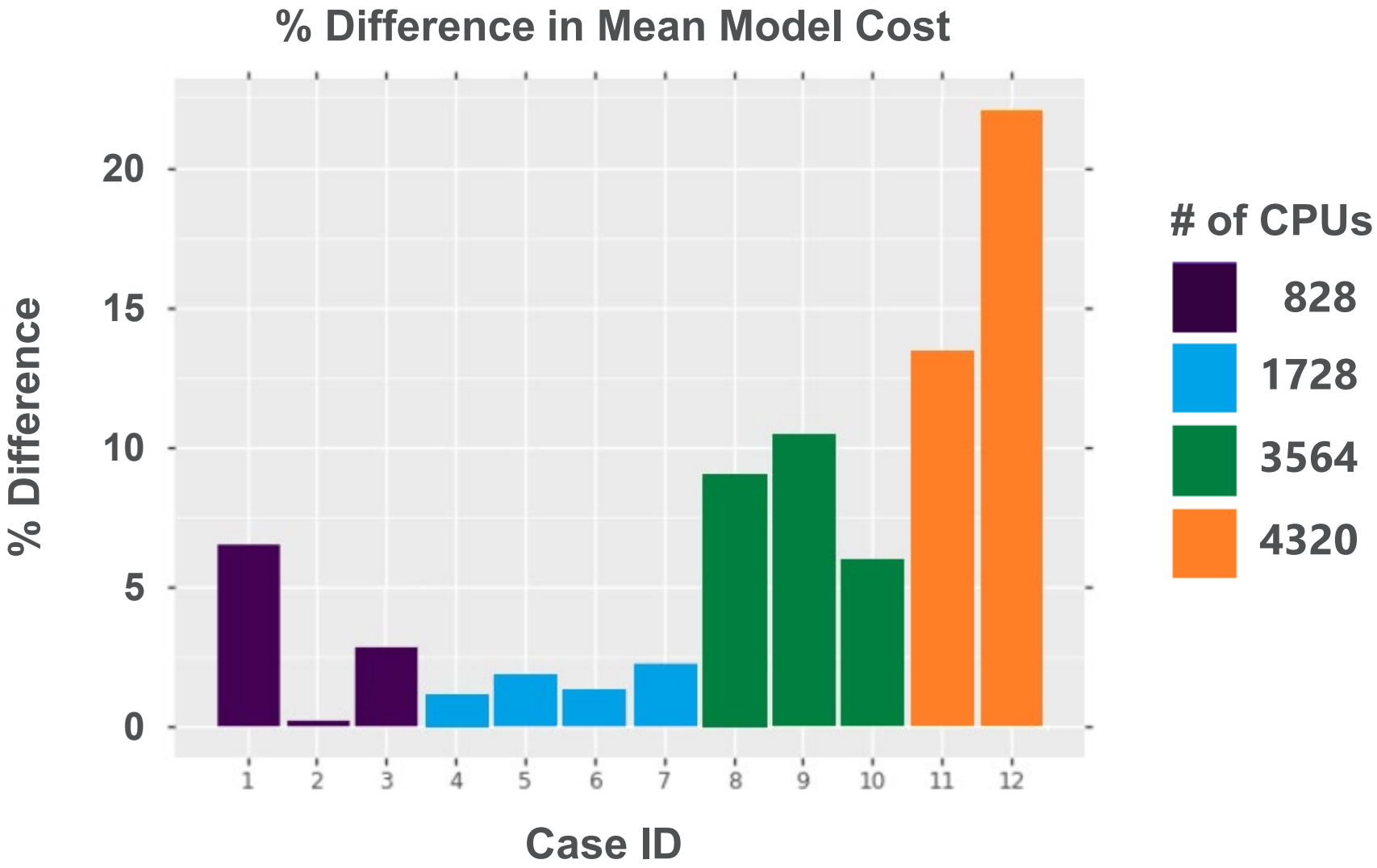


Analysis: Atmospheric Components



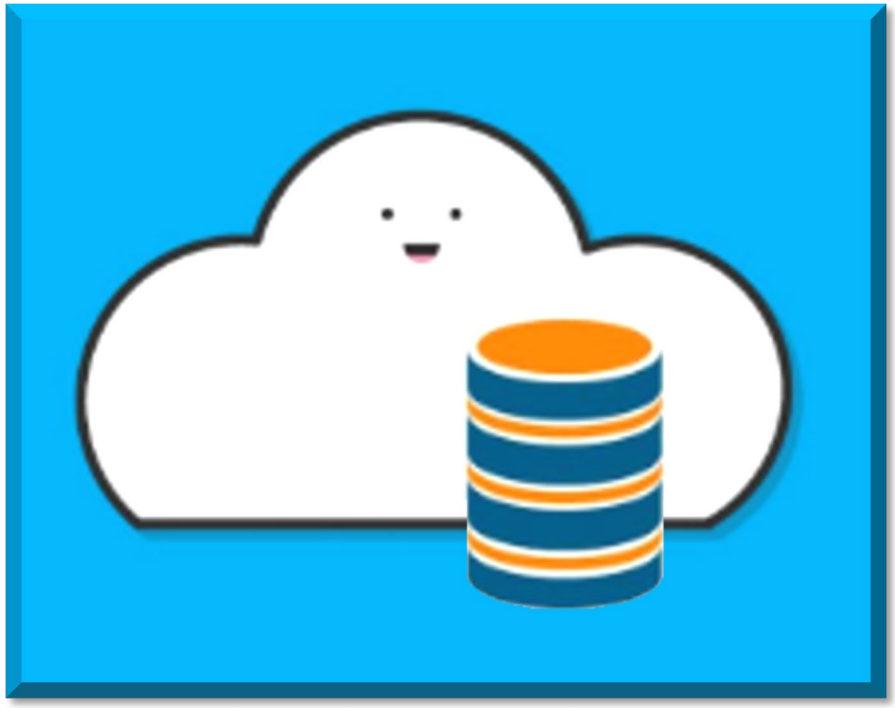
Analysis: System Upgrade

Simulations that span the early July upgrade



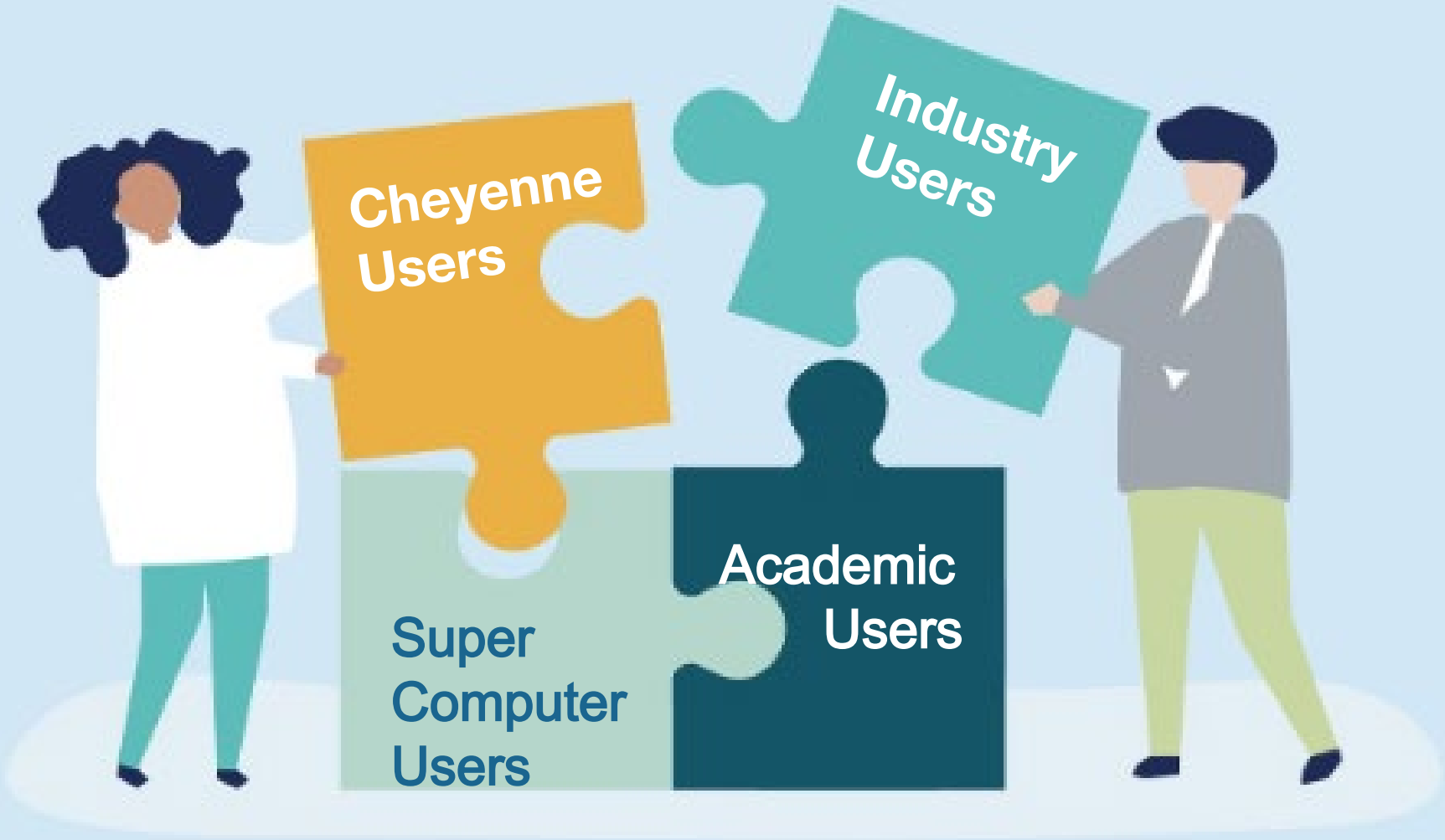
Conclusion

- Useful model of CESM database in the cloud



Conclusion

- Build worldwide CESM database

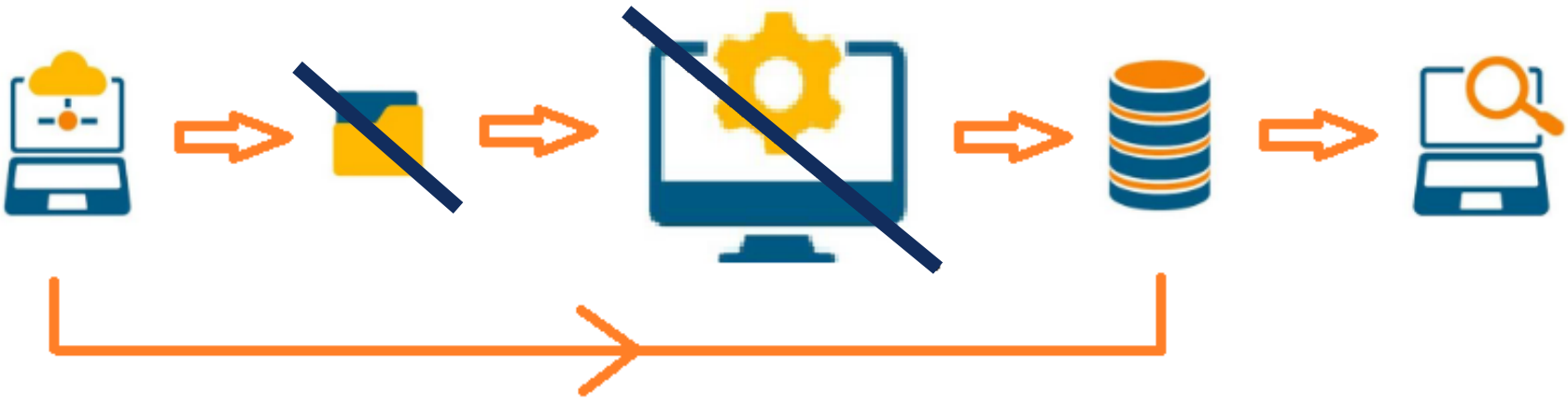


Future Work

Current data capture

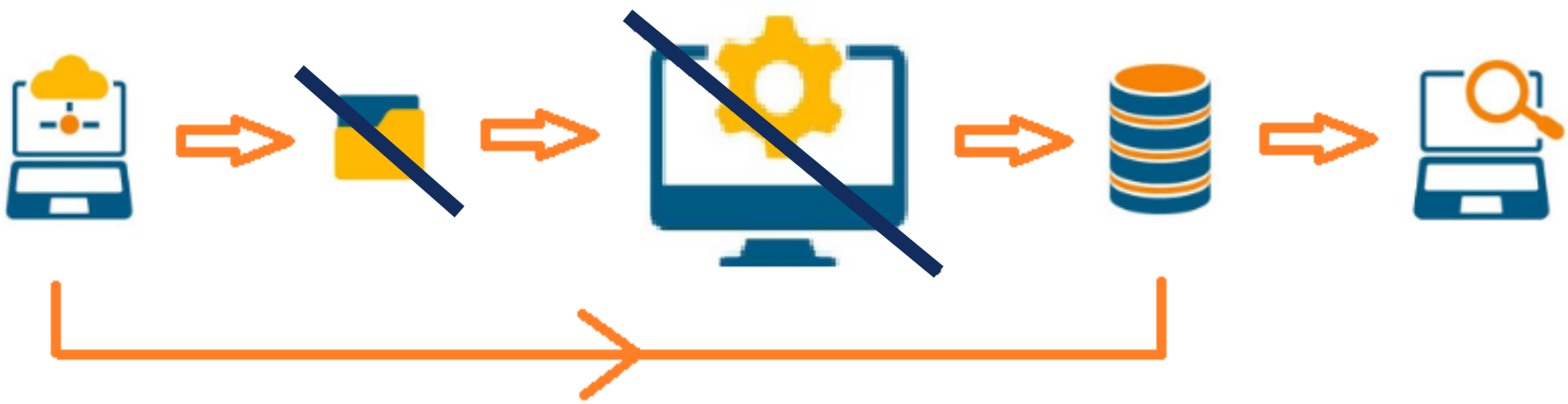
Future Work

Streamline data capture

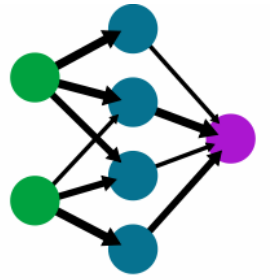


Future Work

Streamline data capture



Feature engineering and machine learning



- Predict performance on configurations
- Verify correct installation and optimal settings for remote users

Acknowledgements

John Dennis
Brian Dobbins
Expert mentors

AJ Lauer
Virginia Do
Expert intern managers

Alice Bertini
SQL Training

References

Balaji, et. al. CPMIP: Measurements of Real Computational Performance of Earth System Models in CMIP6. Geoscience Model Development Issue 10. January 02, 2017. https://www.geosci -model-dev.net/10/19/2017/

Images

Unless otherwise noted, graphics are from www.vecteezy.com

Questions?

Lolita Mannik
lolita@FourWindsPublications.com

This presentation is posted at:
www.FourWindsPublications.com