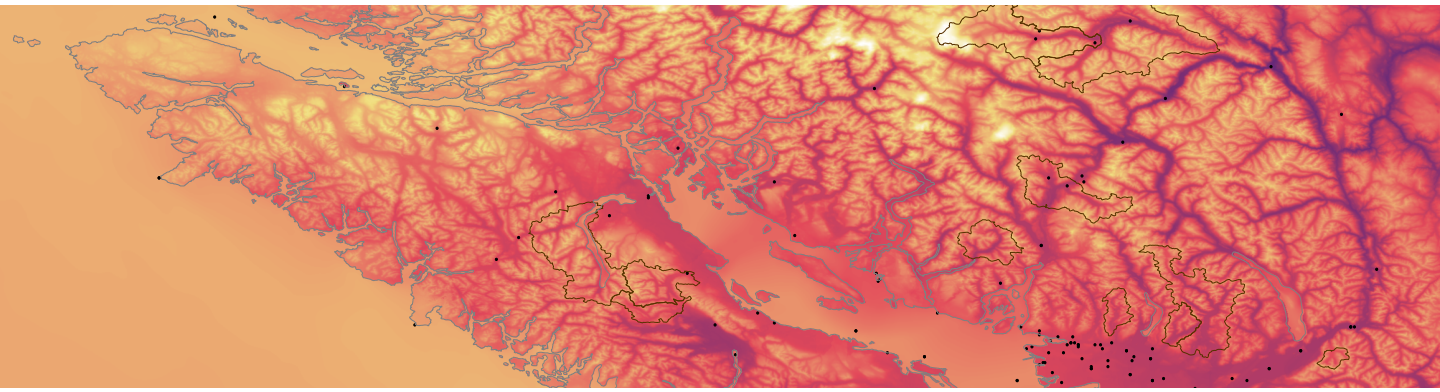

USER MANUAL

WRF/EmWxNet **HIGH RESOLUTION** NOWCASTING SYSTEM

Nadya Moisseeva
nmoisseeva@eos.ubc.ca
August 2015

WRF/EmWxNet High Resolution Nowcasting System User Manual

Nadya Moisseeva

August 2015

Contents

Introduction

WRF/EmWxNet High Resolution Nowcasting System has been developed to produce refined analysis fields by combining model forecast with real-time observations through downscaling and data assimilation. The current system relies on meteorological fields from Weather Research and Forecasting model (WRF) and observations from Emergency Weather Net Database (EmWxNet). This document is intended to provide an overview of the methods as well as user instructions for implementing the nowcasting system operationally.

The first part of the manual, *User Guide*, contains setup instructions, overview of various system configurations and components, as well as a brief description of the output graphics. The second part, *Technical Description*, provides a more detailed review of the methods, analysis and its verification. Lastly, *Future Considerations* chapter overviews the planned developments and improvements.

1 User Guide

1.1 Code Components

Figure 1 shows the various components of the nowcasting system. While the majority of the analysis is performed using Python language (as denoted by *.py file extensions), the primary operational driver and EmWxNet databased wrapper are coded and bash shell (*.bash) and C (*.exe), respectively.

Note, that only one of the components, namely **da_config.py**, is to be modified by the user. The remainder of the system is controlled by the main bash and python drivers automatically. While unnecessary for day-to-day operation of the nowcasting system the primary details and functions of each component are described below.

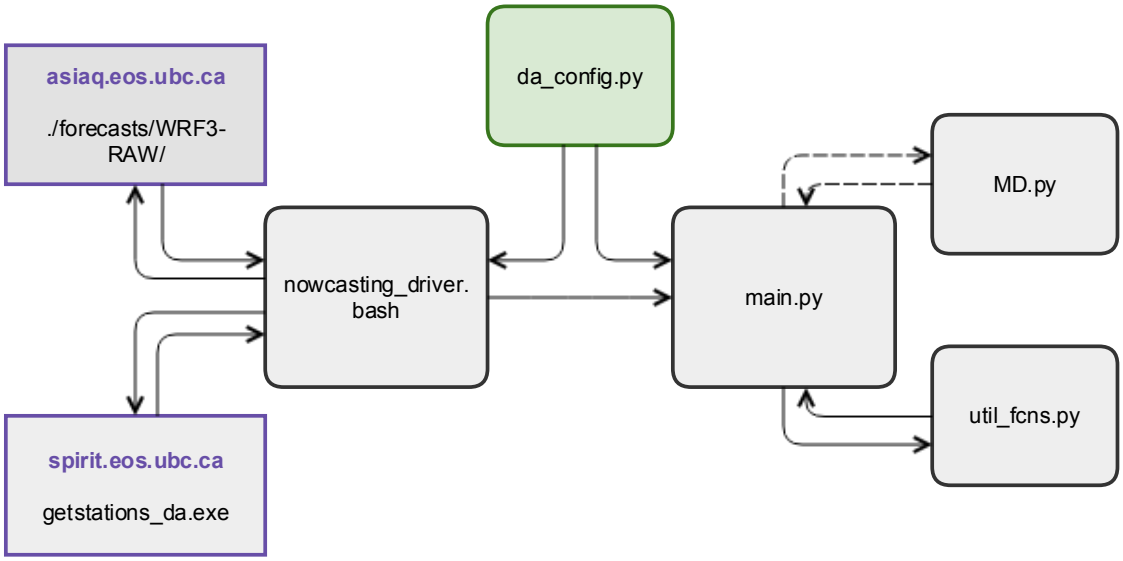

Figure 1: Nowcasting system flowchart. Remote servers typeset in purple; user-controlled component (**da_config.py**) shown in green

da_config.py
Python equivalent of the namelist: contains all of the user-controlled settings. For details of the included variables and parameters refer to Section 1.4.

nowcasting_driver.bash
Bash shell script: drives all of the main system components, including connection to remote servers, moving of files to local computer and initialization of python routines.

- Locates required forecast WRF data; moves raw netcdf files from **asiaq.eos.ubc.ca** server if necessary
- Extracts EmWxNet data from **spirit.eos.ubc.ca** and creates copies on the local computer
- Runs the main Python analysis driver

main.py
Main Python analysis driver:

- Constructs analysis domain (creates basemaps and landmasks, if necessary)
- Interpolates model data to Digital Elevation Model (DEM) resolution
- Performs data downscaling
- Performs data assimilation
- Produces output graphics

May be used as part of an operational system controlled by **nowcasting_driver.bash** or independently.

getstations_da.exe
C-based code wrapper for pulling observations from EmWxNet. Note (!): lat/lon extent of the area for which the observations are extracted is currently hard-coded in the file (includes southern BC only). The domain extent can easily be changed in the input header within the file. It should be equivalent or larger than the domain used for nowcasting (unnecessary stations will be excluded).

MD.py
Mother-Daughter data assimilation script for calculating sharing factors for observation stations. Builds a weights/distances database for observation stations, appends stations as necessary.

util_fcns.py
Python utility module: contains all of the subroutines called by the main Python driver.

1.2 System Requirements

Required languages for the local computer (excludes requirements for remote servers, C-compilers, etc.)

1. **Unix bash shell**. Note (!): GNU-based Linux shell uses a different version of the **date** command, which may not have the same flags as those required by the current operational setup.

2. **Python 2.7**, to be tested with Python 3.4. Primary packages, including NumPy, matplotlib, SciPy

1.3 Setup

This section describes the initial setup of the system. These steps do not need to be repeated for any subsequent new analysis/domain configurations and are only performed once.

1. Create a new folder for storing the code.
   Download the contents of **http://gl.tawhiri.eos.ubc.ca/nmoisseeva/nowcasting** to the new folder.

2. Establish keychain access to **spirit**. Ensure that you are able to ssh and copy files from the server without manual password entry.

3. Mount the **asiaq** forecasts folder as a shared drive (MacOS). For Linux installations, modify the **nowcasting_driver.bash** to use ssh and scp with keychain access.

4. Copy the contents of **./emx** subfolder to **spirit.eos.ubc.ca**. Ensure that the domain specified in the input header of **getstations_da.c** is the same or larger than the one that will be analyzed. Build the C code using the provided **makefile**. A successful compilation will produce a **getstations_da.exe** file.

5. Verify all paths and user names input header of **nowcasting_driver.bash**, save updates.

6. Configure **config_da.py** (refer to Section 1.4) and run analysis (refer to Section 1.5)

1.4 User Configuration

Following the initial setup of the nowcasting system, the domain and analysis configuration can be changed by simply modifying the settings in **da_config.py**. The options and flags along their with their functionality are described below.

- -
#defining input and output locations and source files

fig_dir
Path to folder for storing output graphics

netcdf_dir
Path to storage folder for raw WRF netcdf data from the model.

netcdf_prefix
The name format (without timestamp) of the raw WRF netcdf file.

emx_dir
Path to storage folder for EmWeatherNet data.

geotiff
Name of the GeoTIFF file containing the DEM data for the domain.

landmask_dir
Optional: Path to folder containing numpy landmask.

basemap_dir
Optional: Path to folder containing numpy basemap.

emx_name
Changes not recommended: Name of output file from getstations_da.c. Leave unchanged, unless altered in C code.

- -
#domain configuration

lat
[minimum lat, max latitude]

lon
[min longitude, max longitude]

- -
#data assimilation configuration

run_MD
Flag to use Mother-Daughter approach: 1 for MD data assimilation; 0 for ROI data assimilation

verif_frac
Fraction of data that will be used for model training: the remainder will be used for cross evaluation.

bias_mode
Flag to correct temperature increments: 1 for incremental (bias) correction; 0 for absolute temperature correction

delay_hr
For operational use: analysis delay (in hours)

- -
#ROI DA configuration

roi
Horizontal range of influence for ROI-based data assimilation (in decimal degrees)

elev_roi
Vertical range of influence for ROI-based data assimilation (in meters)

- -
#MD DA configuration

params
a,b,Z_{ref1},Z_{ref2} Parameters for calculating MD sharing factors

dist_cutoff
Maximum anisotropic horizontal distance in degrees (when reached, ending iteration)

weight_cutoff
Minimum change in MD weight values required to continue iteration

- -
#temperature and lapse rate configuration

lvl
Number of model levels to use for lapse rate calculation (starting from surface)

T_range
Fixed colormap range to use for output temperature plots (in C, centered at 0)

1.5 Ways to Run the Analysis

The nowcasting analysis can be performed in an automated as well as manual mode. Usage instructions for both approaches are described below. Note that the first-time run (whether automated or manual) for a new domain/settings may take a substantial amount of time (≈7-10 min), as new basemaps, landmasks and (optionally) MD weights are being calculated. The code is optimized to store the output of these time-consuming operations and allow recycling for subsequent runs.

1.5.1 Operational/Automated Mode

When ran operationally, the flow of the analysis is controlled by **nowcasting_driver.bash** shell script. All the required process flow settings are passed here from **da_config.py** automatically. Based on the user-specified time delay the bash script will copy all the necessary files from remote serves, store them in automatically generated subdirectories (by date) and run the main analysis. Note (!): all timesteps are UTC.

For a single automated run:

1. In terminal, navigate to your local nowcasting folder.
2. Modify **da_config.py** to desired settings.
3. In terminal type: **bash nowcasting_driver.bash**

For a repeated automated run:
Schedule a task to your **crontab** to call **bash nowcasting_driver.bash** with desired frequency

1.5.2 Manual Mode

The Python analysis component (**main.py**) may be ran as a stand-alone code with existing datasets, without evoking connection to remote servers.

For a single manual run:

1. In terminal, navigate to your local nowcasting folder.
2. Modify **da_config.py** to desired setting. Note that in manual mode **delay_hr** variable is ignored.
3. Ensure that raw model and observation data are located in **./year/month/day/hr** subdirectories of **netcdf_dir** and **emx_dir**, respectively.
4. In terminal, type:
   python main.py filename
   where **filename** is the full name of the raw netcdf file, including prefix and timestamp

1.6 Output Graphics

The final product of the analysis consists of high resolution field plots, verification graphs, as well as several auxiliary graphics. These are stored in **./year/month/day/hr** subdirectory within the user-specified **fig_dir**. Sample images and their brief descriptions are shown below.

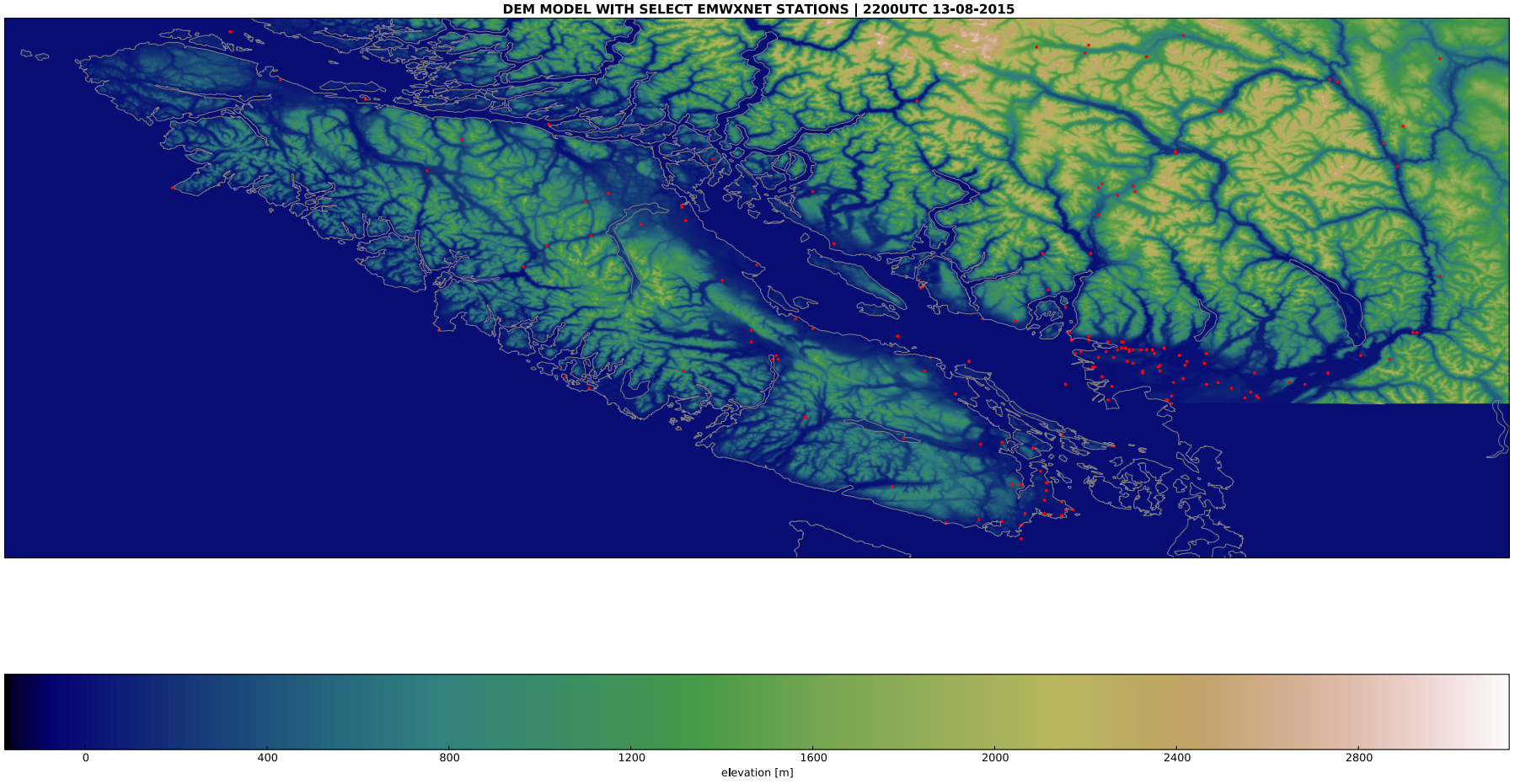

Figure 2: Analysis domain with 12arcsec DEM data. Select EmWxNet stations with observations for 2015-08-13 2200 UTC shown as red scatter.

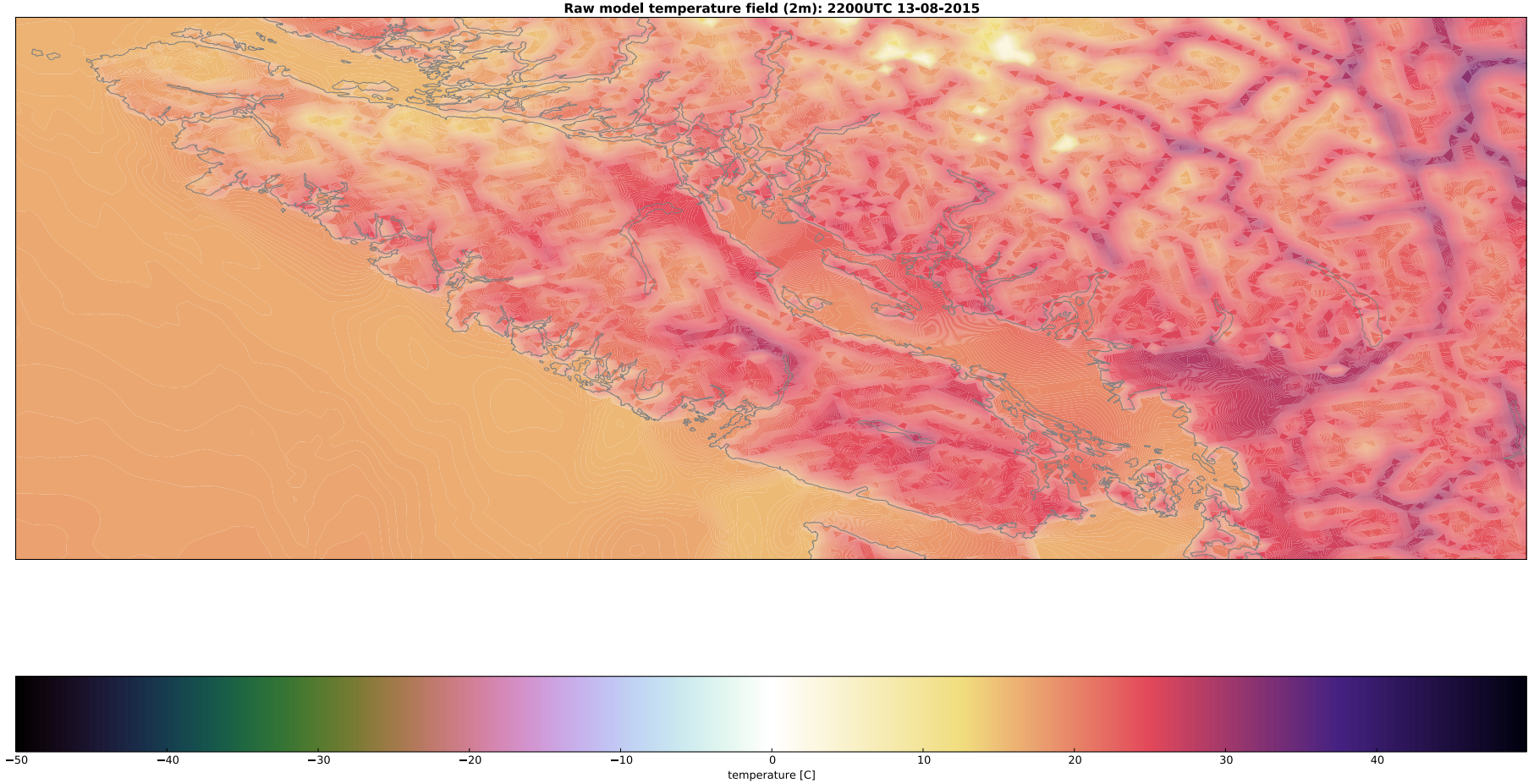

Figure 3: Raw 2m temperature fields from WRF netcdf output (4km resolution).

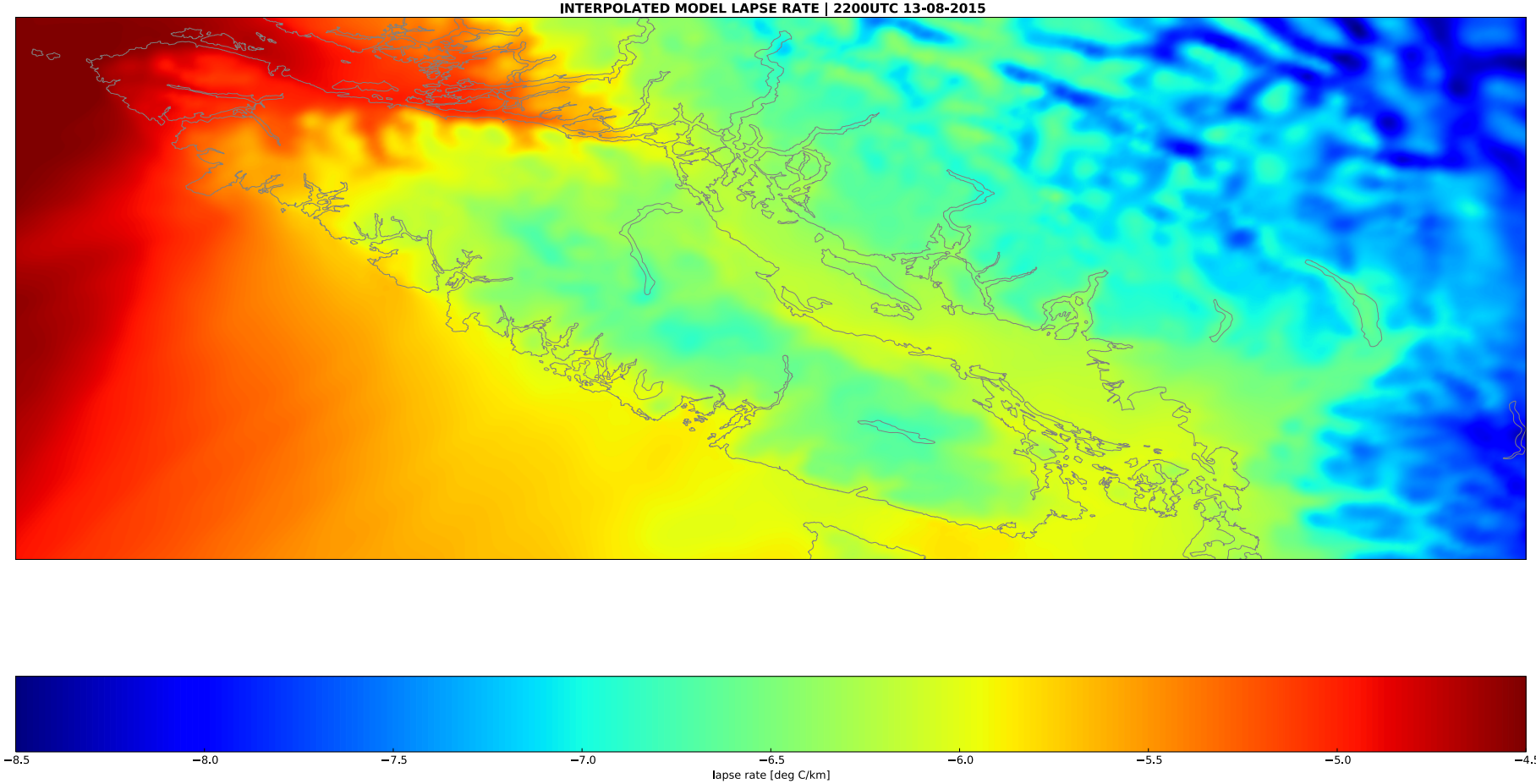

Figure 4: Raw WRF model lapse rate, based on first 20 layers about the surface.

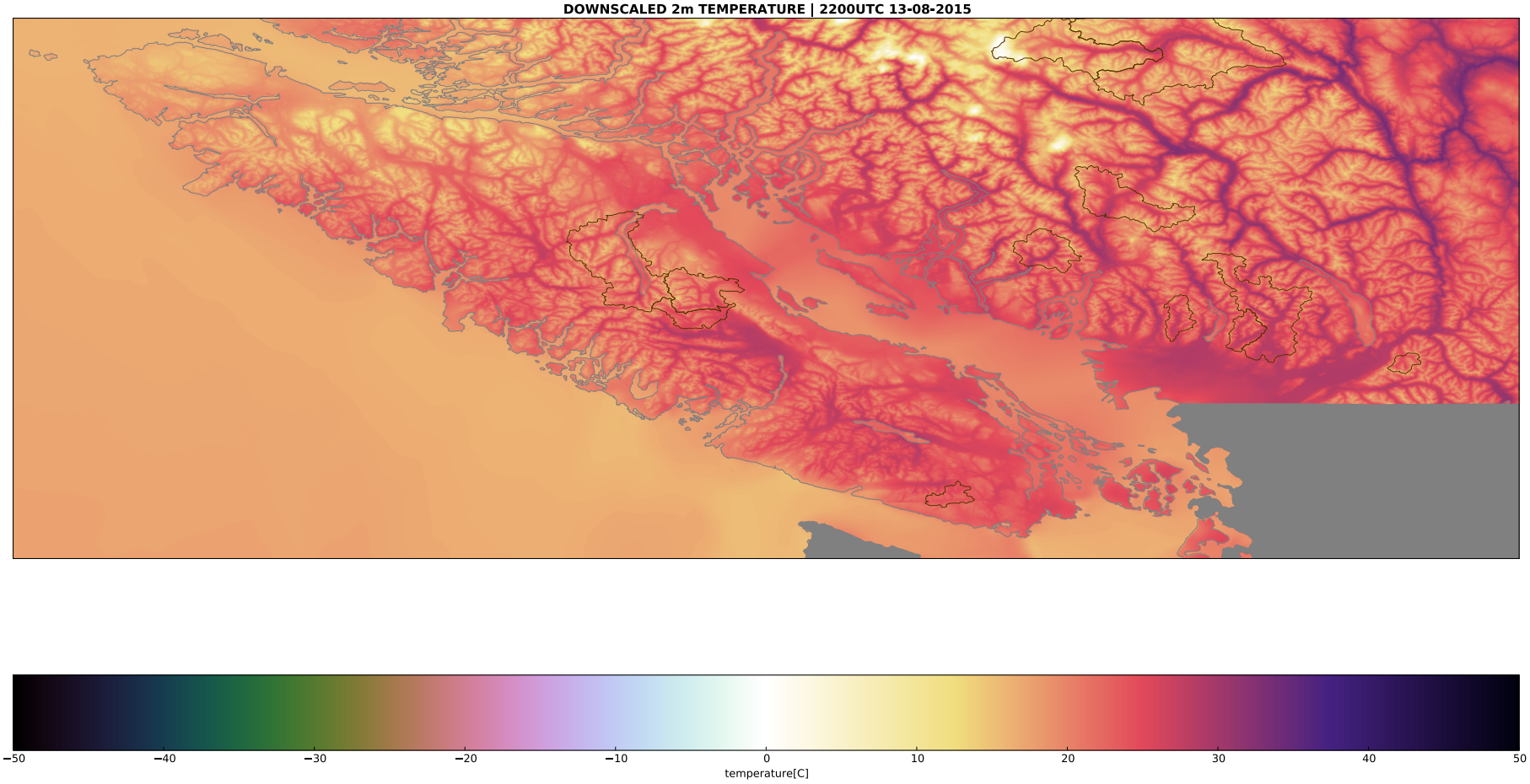

Figure 5: Downscaled 2m temperature. Gray region masks US territory (no available DEM data). Fine black contours mark BC Hydro watersheds, as designated by Major Hydro Watersheds Project.

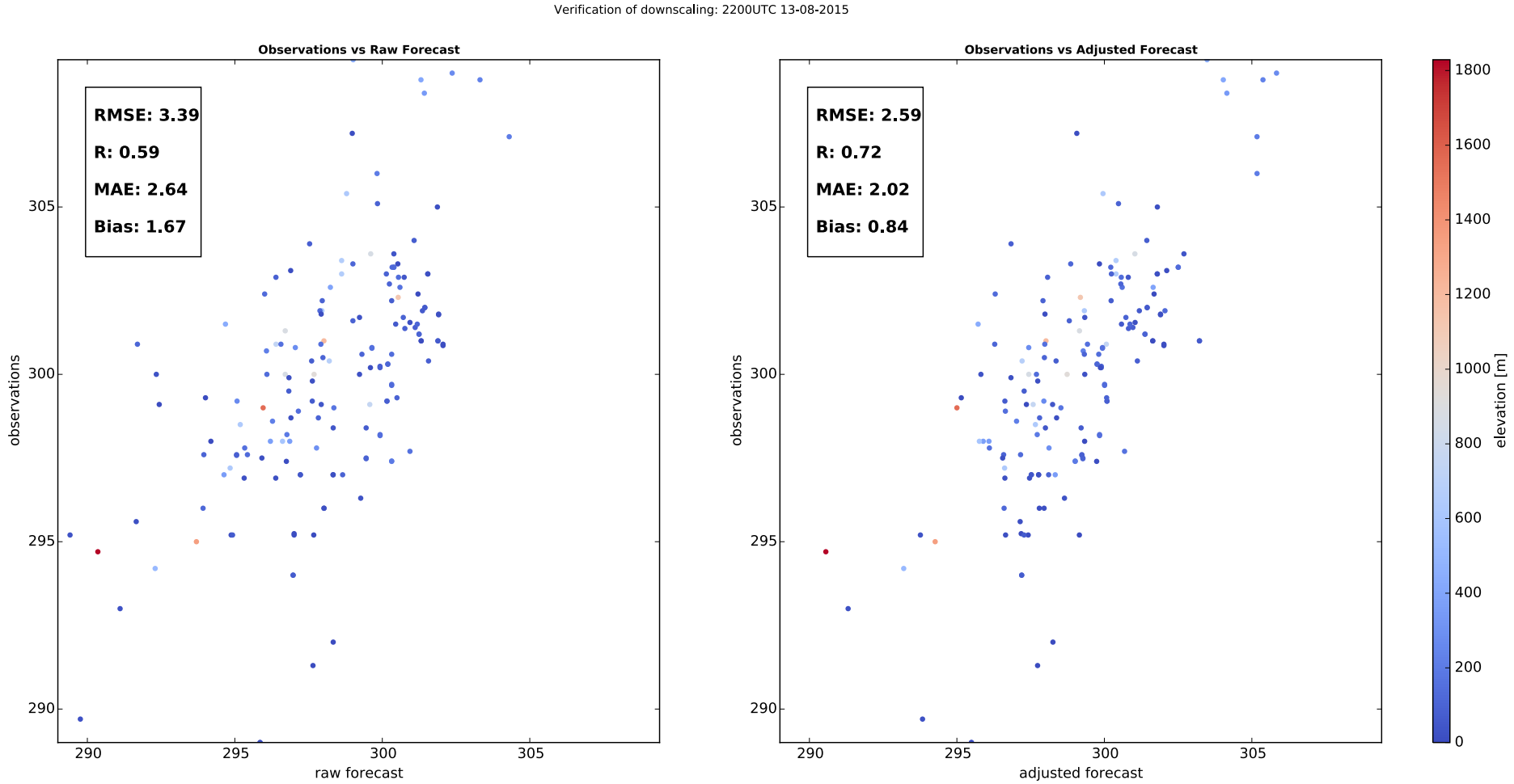

Figure 6: Downscaling evaluation scatter graphs shown for Raw vs. Observed and Downscaled vs. Observed. Marker colors correspond to the elevation of observation stations. Also shown: root mean squared error (RMSE), correlation coefficient (R), mean absolute error (MAE) and bias.

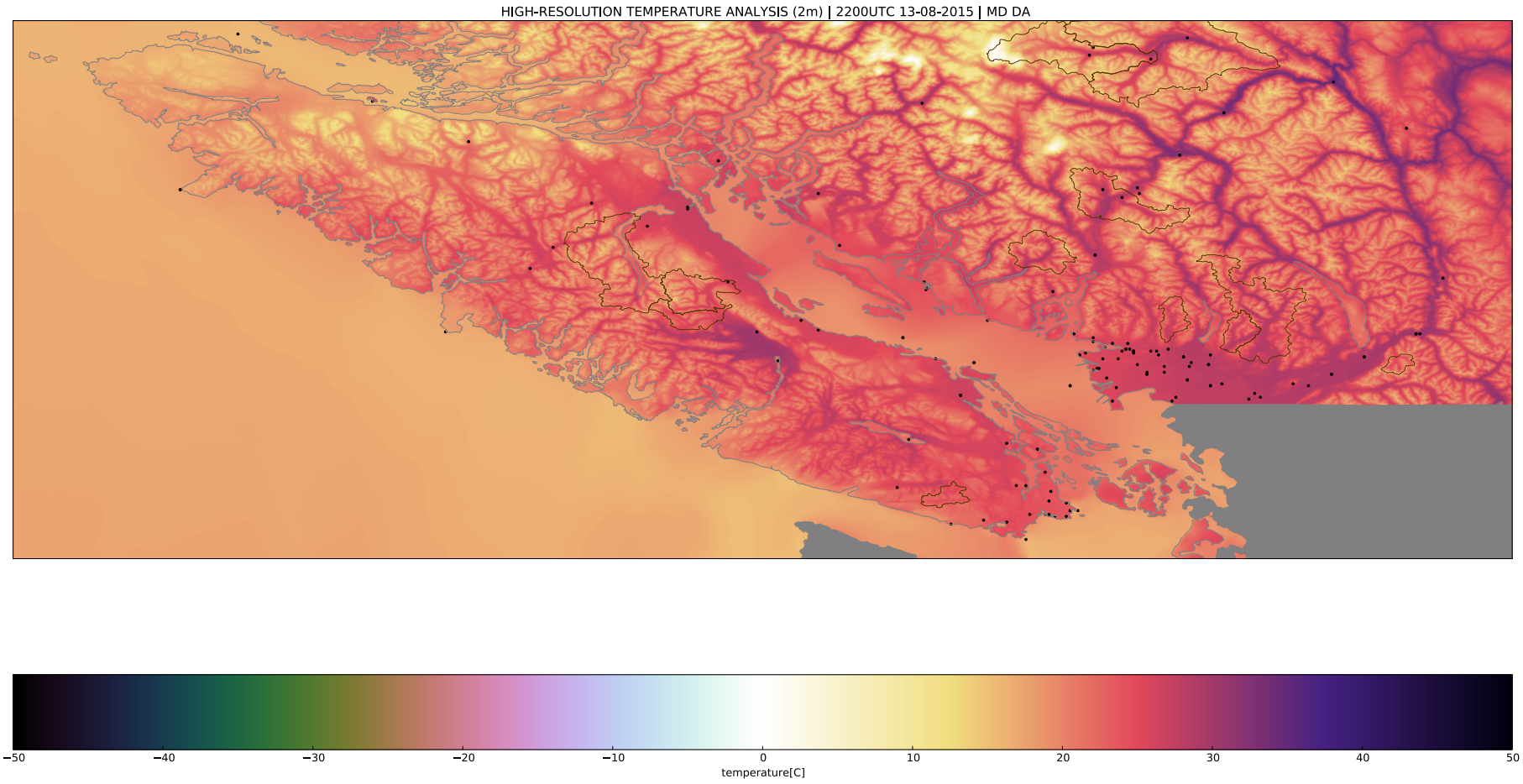

Figure 7: Corrected 2m temperature, based on MD data assimilation approach. Gray region masks US territory (no available DEM data). Fine black contours mark BC Hydro watersheds, as designated by Major Hydro Watersheds Project.

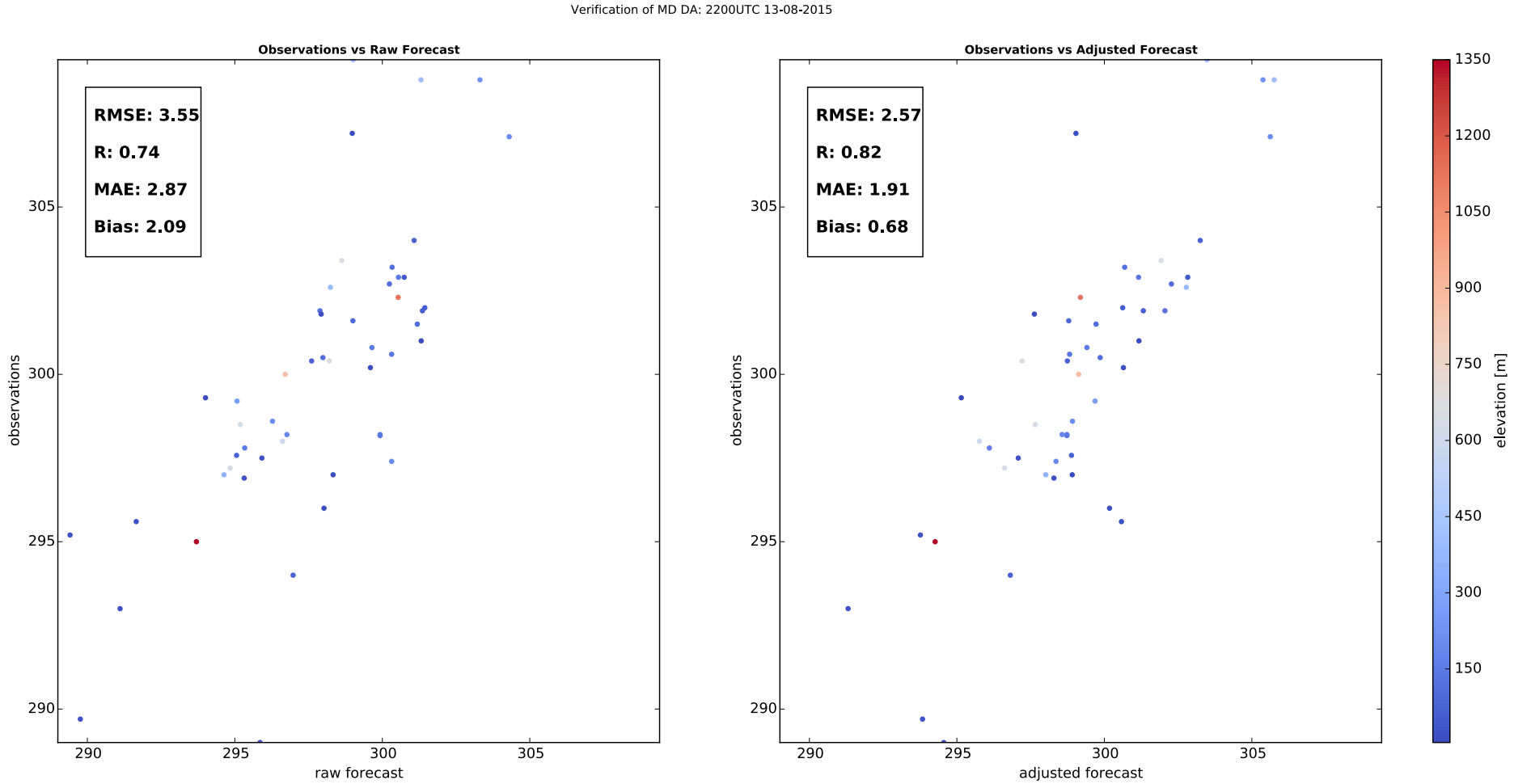

Figure 8: Cross-evaluation of high-resolution analysis (downscaling + data assimilation) using an independent test dataset. Raw vs. Observed and Corrected vs. Observed. Marker colors correspond to the elevation of observation stations. Also shown: root mean squared error (RMSE), correlation coefficient (R), mean absolute error (MAE) and bias.

2 Technical Description

The *Technical Description* provides a brief summary of the methods as well as a scientific rationale for their implementation. More importantly, it aims to explicitly highlight the underlaying simplifications and the potential limitations of WRF/EmWxNet High Resolution Nowcasting System design. The current analysis is limited to surface temperatures only, however, will be expanded to include precipitation and wind fields in the future.

2.1 Data and Domain Setup

DEM data
High resolution (12arcsec, or ≈350m) DEM data were obtained from Natural Resources Canada Online Geospatial Data Extraction tool [1], available at **http://geogratis.gc.ca/site/eng/extraction**. The nowcasting system relies on the provided GeoTIFF format (georeferenced raster image in Geographic coordinate system).

WRF model forecast fields
The gridded first guess temperature as well as lapse rate fields are produced from raw (non-postprocessed) WRF3-ARW NAM model runs, initialized daily at 0000UTC with NCEP data. The tested configuration used **d03** domain with 4km resolution.

EmWxNet observational data
During each initialization of the analysis for a given forecast hour fresh observation data are pulled from the EmWxNet database. It is assumed that with any additional time more data could have been ingested into the database, which may improve the analysis (exception: no new EmWxNet data are loaded when system is ran in manual mode). All stations that fall within the user-specified domain bounds (excluding US territory) are extracted. The data are subsequently filtered to include only those measurements that fall within 5 minutes of the nowcast hour. If the allowed time range is extended, artifacts may be introduced into the analysis fields as neighboring stations reporting observations from opposite ends of the tolerance interval may produce conflicting readings.

The observation stations are then quality-controlled by ensuring that the reported temperatures fall within a realistic range between -50 and +60 degrees C. Lastly, the reliability of station metadata is tested by locating the DEM grid point closest to the listed station location and ensuring that the elevation discrepancy between the metadata altitude and DEM grid does not exceed 200m (typically 4-5 stations are excluded for the tested domain).

2.2 Downscaling Method

Coarse (4km) first-guess temperature field (Figure 3), produced from diagnostic **T2** variable in WRF output, are first interpolated to high resolution DEM grid. Due to frequent sharp gradients in temperature between land and water and the discrepancy between DEM and model coastlines, straight forward interpolation is insufficient and results in numerous artifacts and edge effects. Consequently, a split-interpolation solution is adopted. High-resolution landmask is generated for the given domain by classifying DEM points as land/water using the Python basemaps coastal polygons objects. Low-resolution model landmask is used to categorize model gridpoints. The latter are then interpolated separately to high-resolution grid, each spanning the entire DEM domain. The resultant fields are then masked with high resolution DEM landmask (removing water in land interpolation, and land in water interpolation) and concatenated together to produce a single high-resolution interpolated field.

The next step is to interpolate model terrain elevation to high-resolution DEM grid. Since terrain smoothing is assumed to be applied to the majority of model runs no additional filtering is performed.

Surface lapse rate is then calculated for each model grid point, using the user-specified number of model levels. A profile is constructed using temperature and level hight and profile slope is calculated for each grid point. During the course of the analysis several approaches were tested (using all surface temperature points from the model to produce a temperature profile; using observation temperatures only; subdividing

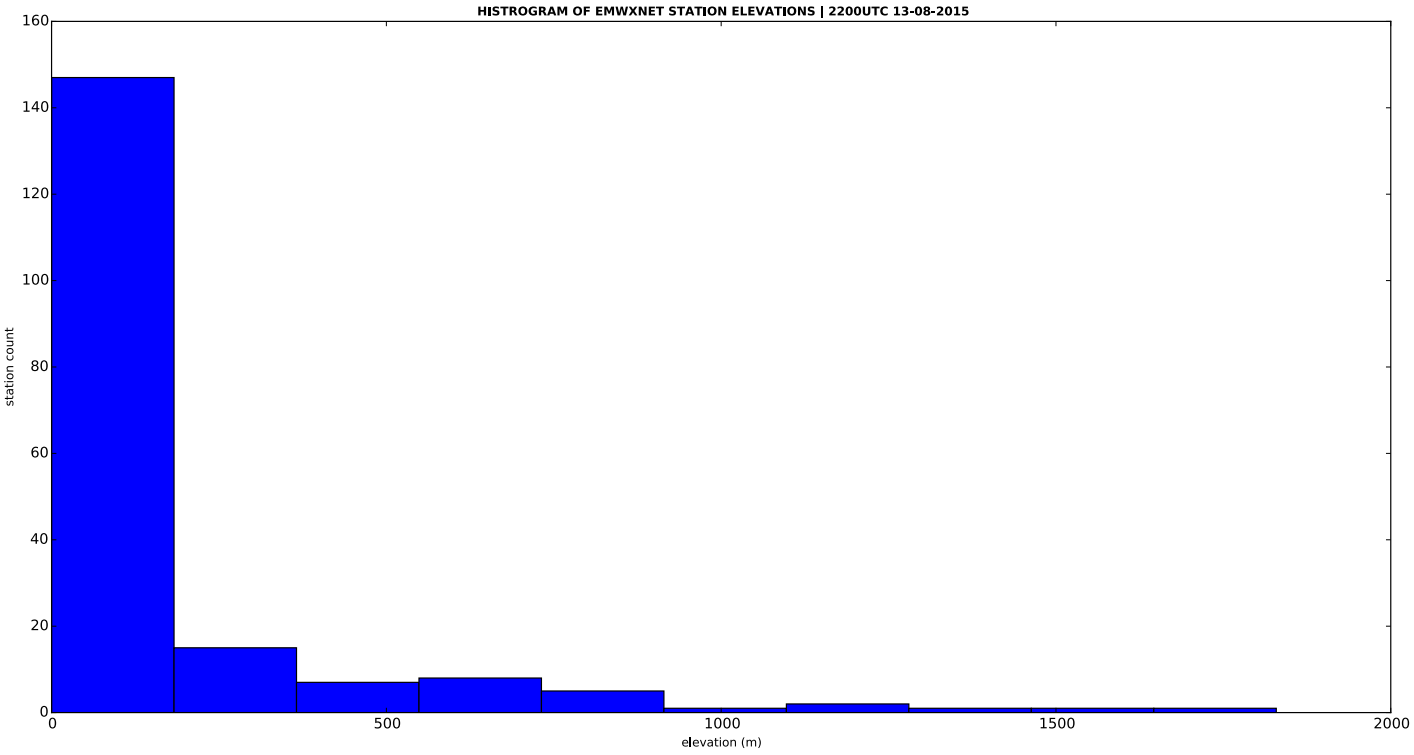

Figure 9: Histogram of station elevations.

the domain and several others). The alternatives fail to account for the high spatial variability in lapse rates across the domain. As can be seen in Figure 4, areas shielded by mountains often show values close to dry lapse rate, while over-the-water and coastal regions have a substantially 'wetter' lapse rate.

Another advantage of this method is that it allows to partially account for local effects such as cold air pooling and temperature inversions. Generally, inversion is unlikely to cause issues in data assimilation as the vast majority of observation stations is located at near ocean-level (Figure 9). Similarly to height, the calculated model lapse rate field is interpolated to DEM grid.

Lastly, the high-resolution interpolated temperature field is lapse-rate corrected, using the steps below for each DEM grid point:

$$\begin{aligned} \Delta h &= z_{DEM} - z_{interp} \\ \Delta T &= \Delta h * \gamma_{interp}/1000 \\ T_{final} &= T_{interp} + \Delta T \end{aligned} \tag{1}$$

where z, T and γ are elevation, temperature and lapse rate, respectively.

The resultant high-resolution temperature fields can be seen in Figure 5.

2.3 Data Assimilation Methods

EmWxNet observations can be incorporated into the downscaled fields using two different methods: based on Region of Influence (ROI) and using Mother-Daughter (MD) approach. Both can be used to nudge the downscaled model first guess temperatures towards observations by either spreading the absolute observation value to nearby grids or by spreading the bias (or increment). Hence, four different configurations are

possible. Based on a preliminary (and rather qualitative) assessment the ROI method tends to perform better products in incremental mode, while MD assimilation scores higher by spreading the absolute observed value. Further, long-term testing would be necessary to determine, whether this assumption for all times of day/month/season.

2.3.1 Region of Influence (ROI)

One of the biggest advantages of the ROI method is that it is not computationally expensive. The current configuration produces a domain with nearly two million grid points and approximately 200 observation stations - sufficient to substantially increase the nowcast time delay. While nearest-neighbor search on such a large domain is typically a difficult computational task, the current analysis uses multiple passes of KD-tree algorithm [3] to identify grids affected by observations, based on user-defined horizontal region of influence. Once the grid points are identified the temperature field is corrected using a basic inverse-distance weighing scheme in three dimensions (using user-specified vertical ROI for elevation adjustment).

In incremental mode:
Let X be 2-dimensional vector corresponding to a grid point location. Then for each grid point $X_i = (x_i, y_i)$, affected by one or more observation stations at $X_j = (x_j, y_j)$, we can calculate a weighted adjustment. In the horizontal:

$$Dh_{ij} = |X_i - X_j|$$
$$Wh_{ij} = (\frac{(r - Dh_{ij})}{r})^2$$

where Dh, Wh and r are horizontal distance, horizontal weight and user-defined horizontal radius of influence, respectively.

In the verital:

$$Dv_{ij} = |z_f(x_i, y_i) - z_{obs}(x_j, y_j)|$$
$$Wv_{ij} = (\frac{(r_e - Dv_{ij})}{r_e})^2$$

where Dv, Wv, z and r_e denote vertical distance, vertical weight, elevation, and user-defined vertical ROI tolerance, respectively.

The total weight Wt_{ij} is simply:

$$Wt_{ij} = Wh_{ij} * Wv_{ij} \tag{2}$$

If T is the 2m temperature and subscripts obs and f denote values from observation stations and downscaled forecast, we can compute the final adjusted grid point value $T_i(x_i, y_i)$ by recursively correcting the forecast with a weighed bias for each station j.

$$\Delta T_{ij} = T_{obs}(x_j, y_j) - T_f(x_j, y_j)$$
$$T_i = T_f(x_i, y_i) + \Delta T_{ij} * Wt_{ij} \tag{3}$$

In temperature-adjustment mode:
Alternatively, rather than correcting temperature increments, it is possible to spread lapse-rate adjusted observation temperature values. Using Eq. 2 to calculate the total weight and model lapse rate γ_{interp} to calculate T_i:

$$T_i = T_f(x_i, y_i) * (1 - Wt) + (T_{obs}(x_j, y_j) + \frac{Dh_{ij} * \gamma_i(x_i, y_i)}{1000}) * Wt \tag{4}$$

The main drawback of this approach is that for days with a notable discrepancy between model and observations (i.e. for "bad" forecasts) the correction is prone to producing a so-called bull's eye effect. While statistically the verification scores are still likely to show improvement, the output fields will present an unphysical visual pattern. In such cases the Mother-Daughter approach discussed next is, without a doubt, the superior choice for data assimilation.

2.3.2 Mother-Daughter (MD)

The utility of the MD approach was first demonstrated for surface temperature analyses by Deng ([2]), and was shown to outperform other schemes over mountainous and coastal terrain. The main challenge with this method is the computational load of calculating sharing factors for the large number of stations in high resolution grid space. As mentioned earlier, in an attempt to mitigate this problem the current system builds a database of station sharing factors and anisotropic distances for each user-defined set of MD parameters, and appends them as necessary. The weight adjustment W_{ij} for grid point i due to observation station j is calculated as follows:

$$W_{ij} = (\frac{D_{max} - D_{ij}}{D_{max}})^2 * S_{ij} \tag{5}$$

where D_{max} is the user-defined maximum distance cutoff (**cutoff_dist**), D_{ij} is the MD anisotropic distance to from X_j to X_i and S_{ij} is the MD sharing factor. An example of the spread of influence of a subset of observation stations can be seen in Figure 10.

The temperature field is then adjusted in either incremental or absolute mode. Similarly to ROI method to produce T_{ij}, bias is simply added to the forecast temperature field, while in absolute mode the observation temperature is lapse-rate adjusted to each grid-point height. Lastly, to compute the combined effects of all stations in MD mode for grid point i:

$$T_i = \frac{(1 * T_f(x_i, y_i) + \sum_{j=1}^{n} \Delta T_{ij} * W_{ij})}{1 + \sum_{j=1}^{n} W_{ij}} \tag{6}$$

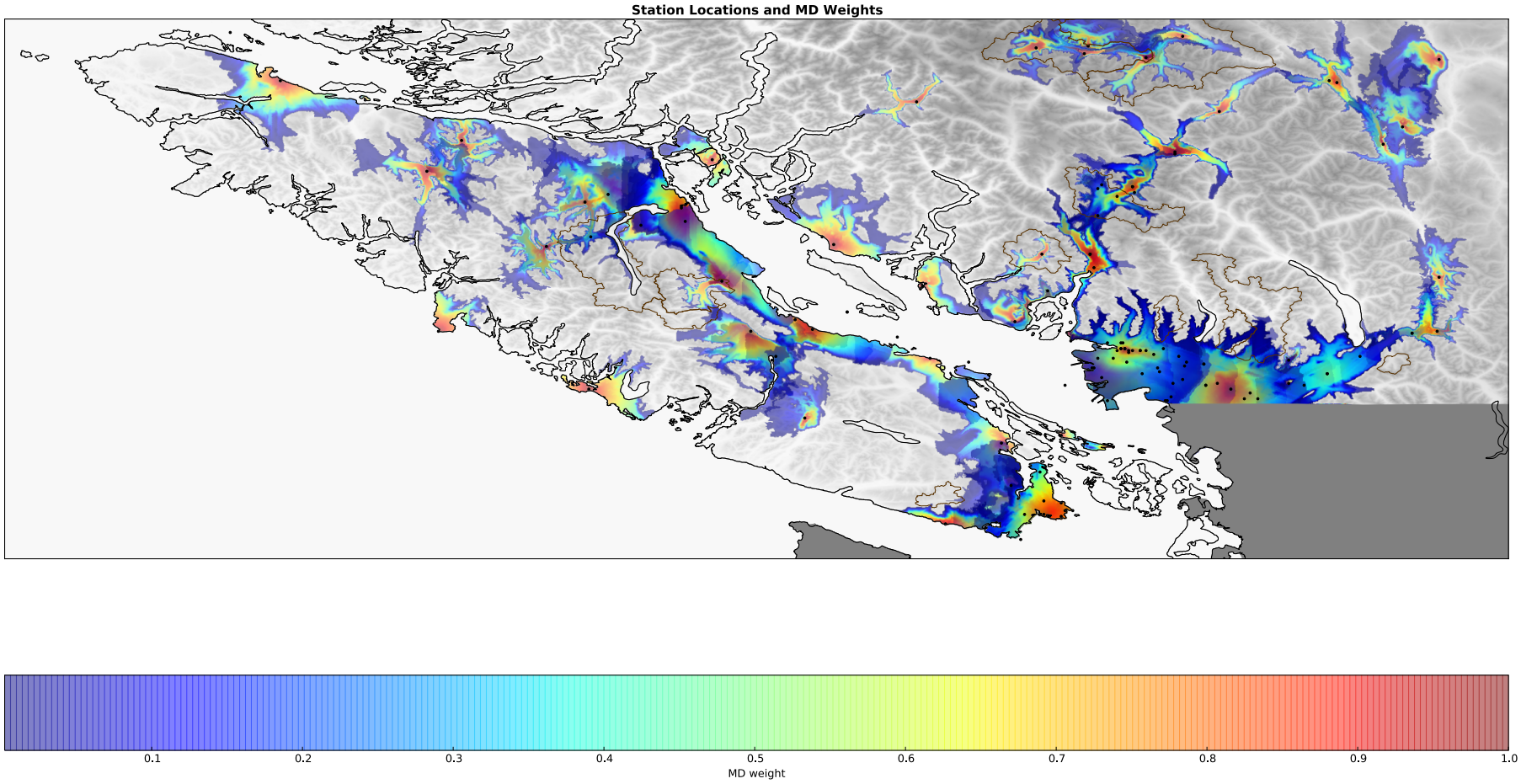

Figure 10: Observation stations and the strength of their influence on surrounding gridpoints‘ (shown for a subset of stations only)

2.4 Verification

Evaluation of each run is embedded in the code and is performed for both downscaling and data assimilation analysis components. Any on-water points are excluded from evaluation to avoid edge effects and errors due to land/water classification differences between the model, DEM grid and basemaps. Figure 6 is an example of standard graphical output for downscaling verification. All available land points are used for the scatter plots and statistical metric calculations. Generally, for days with fairly accurate first-fuess forecast fields, downscaling alone produces a measurable improvement in verification statistics.

Final analysis fields produced through downscaling and data assimilation are assessed using cross-evaluation with an independent test dataset. Corrections are performed for a user-defined fraction of stations, randomly selected from available observation locations (training subset), and subsequently evaluated with the remaining subset of test stations. Statistical metrics included in the output plot (Figure 8) are similarly based on the test data only. Typically, the most notable consistent improvements are seen in bias ($\approx$70%) and MAE ($\approx$50%). RMSE improvements typically vary between $\approx$30-70%. Note that these are rough approximations only based primarily on summer, daytime runs.

3 Future Developments

The current system provides a basic foundation for further developments in all: scope, methodology, computation, evaluation, and graphics. This section aims to summarize the various planned extensions and upgrades and provide an approximate timeline for their implementation.

3.1 Extended Scope

One of the greatest priorities is to extend the nowcasting analysis to other meteorological variables. The current goal is to include wind and precipitation fields. While the technical implementation of downscaling and data assimilation will largely overlap with the code already used for temperature, the relevance and appropriateness of the methods must be reevaluated for each new variable. *Projected completion: October 2015*

3.2 Technical Considerations

The ability to improve the nowcasting system depends largely on identifying the particular weaknesses of the methodology. While the evaluation statistics, including bias, MAE, RMSE and correlation, are currently calculated for each run there is no long-term performance trend analysis. Once the nowcasting system is running continuously in operational mode, it would be possible to harvest and archive the statistics for later analysis. *Projected completion: November 2015*

The nowcasting code has not been tested with Python 3.4 (current version 2.7). *Projected completion: November 2015*

3.3 Methodology Considerations

Minimizing the time delay between the nowcast hour and finalized analysis products is central to ensuring this system is useful to the forecasters. While the time lag is inevitable (hence, making the analysis a backcast), persistence is likely to make the backcast information useful for more current first-guess fields. A possible development for this work is to assimilate incremental field corrections from the backcast (bias) into later forecasts with a short relaxation time towards model fields. This may allow the system to become a "true" nowcast. The usefulness of this persistence assimilation could be assessed once continuous evaluation statistics are harvested. *Projected completion: December 2015*

3.4 Improvements in Graphical Outputs

The current goals for graphical output is to allow the user to select from several output map projections (currently only lat/long is supported) and to develop several custom colormaps for output plots. *Projected completion: September 2015*

References

[1] Natural Resources Canada. Geospatial data extraction. [available at]http://docs.scipy.org/doc/scipy-0.14.0/reference/generated/scipy.spatial.KDTree.html.

[2] X. Deng and R. Stull. A mesoscale analysis method for surface potential temperature in mountainous and coastal terrain. 133:p389–408, 2005.

[3] Scipy Community, [available at]http://docs.scipy.org/doc/scipy-0.14.0/reference/generated/scipy.spatial.KDTree.html. *SciPy v0.14.0 Reference Guide - KD Trees*, 2014.