

1 Overview

This code consists of three parts: the Radar Filtering (**RF**), Karhunen–Loève Expansion (**KL**), and Hidden Markov Model (**HMM**). **RF** takes raw radar (SAR) data and applies a Bayesian filter in both time and space, **KL** takes raw optical (EVI) data and projects it into the residual space of a truncated KL expansion to get anomaly values, and **HMM** uses the filtered radar data and anomaly values to detect deforestation. This means that the **RF** and **KL** code must be run first to generate the inputs for the **HMM** code. The order in which the first two are run doesn't matter. There is also a Validation section, which creates the random optical day removal figures.

2 File Structure

Figure 1 shows the file structure for the code. In the first level we have the `/Toolboxes/`, `/Source/`, and `/Data/` folders, as well as the `paths.m` file, which sets the directory.

- `/Toolboxes/` contains code taken from elsewhere, in this case just the `Laps.m` file from the BLOPEX package (<https://bitbucket.org/joseroman/blopex/src/master/>). This file is used by **RF**.
- `/Source/` contains all the code written for this project. `/Source/Initialization/` contains the initialization files for **RF**, **KL** and **HMM**. `/Source/Run/` contains the respective run files. `/Source/Validation/`, as mentioned before, contains the code to generate the random optical day removal figures. `/Source/RF_Modules/`, `/Source/KL_Modules/` and `/Source/HMM_Modules/` contain the modules for their respective parts of the code. These do not need to be modified and so the individual files are not listed in Figure 1. `/Source/KL_Modules/Fill_NaN/` contains optional files that handle missing optical data. These are not needed to recreate the results in the paper, however some preliminary results indicate that they can improve performance. They will be described in Section 3.2.
- `/Data` contains `/Data/Input/` and `/Data/Output/`, which contain the input and output folders for **RF**, **KL**, **HMM**, and the Validation code. The files listed in `/Data/Input/` are what are needed to recreate the results for the full 92 km region in the paper, as well as the random optical day removal figures. This data can be found at https://www.dropbox.com/scl/fo/necxucp50n8yd7jr1u0zg/APiyR_ScRBNGnYP-b5QHRtA?rlkey=xh706o5gi44gxic5ajlwxxz97m&e=1&st=x5mj36k8&dl=0

3 Initialization Files

Before running the code the input data file paths, folder paths, and a number of parameters must be set in the initialization files in `/Source/Initialization/`. These requirements are described in the following subsections.

3.1 RF Initialization

`RF_Initialize.m` requires the file path for the raw radar data, in this case `/Data/Input/RF/S1.mat`, and the **RF** output folder path `/Data/Output/RF/`. The **RF** parameters are listed in Table 1.

Table 1: Initialization parameters for **RF**

Parameter Name	Parameter Description
C	Spatial smoothing penalty $(\sigma_1/\sigma_2)^2$ (See Supplemental Section S.II)
C2	Temporal smoothing penalty $(\sigma_1/\sigma_3)^2$ (See Supplemental Section S.II)
pcg.maxit	Maximum number of iterations allowed while doing PCG
pcg.tol	Error tolerance/convergence threshold for PCG

See <https://www.mathworks.com/help/matlab/ref/pcg.html> for further details on the PCG parameters.

3.2 KL Initialization

`KL_Initialize.m` requires the file paths for the raw optical data and the corresponding list of days when this data was recorded, which are `/Data/Input/KL/S2.mat` and `/Data/Input/KL/S2_Days.mat` respectively, as well as the **KL** output folder path `/Data/Output/KL/`. The **KL** parameters are listed in Table 2.

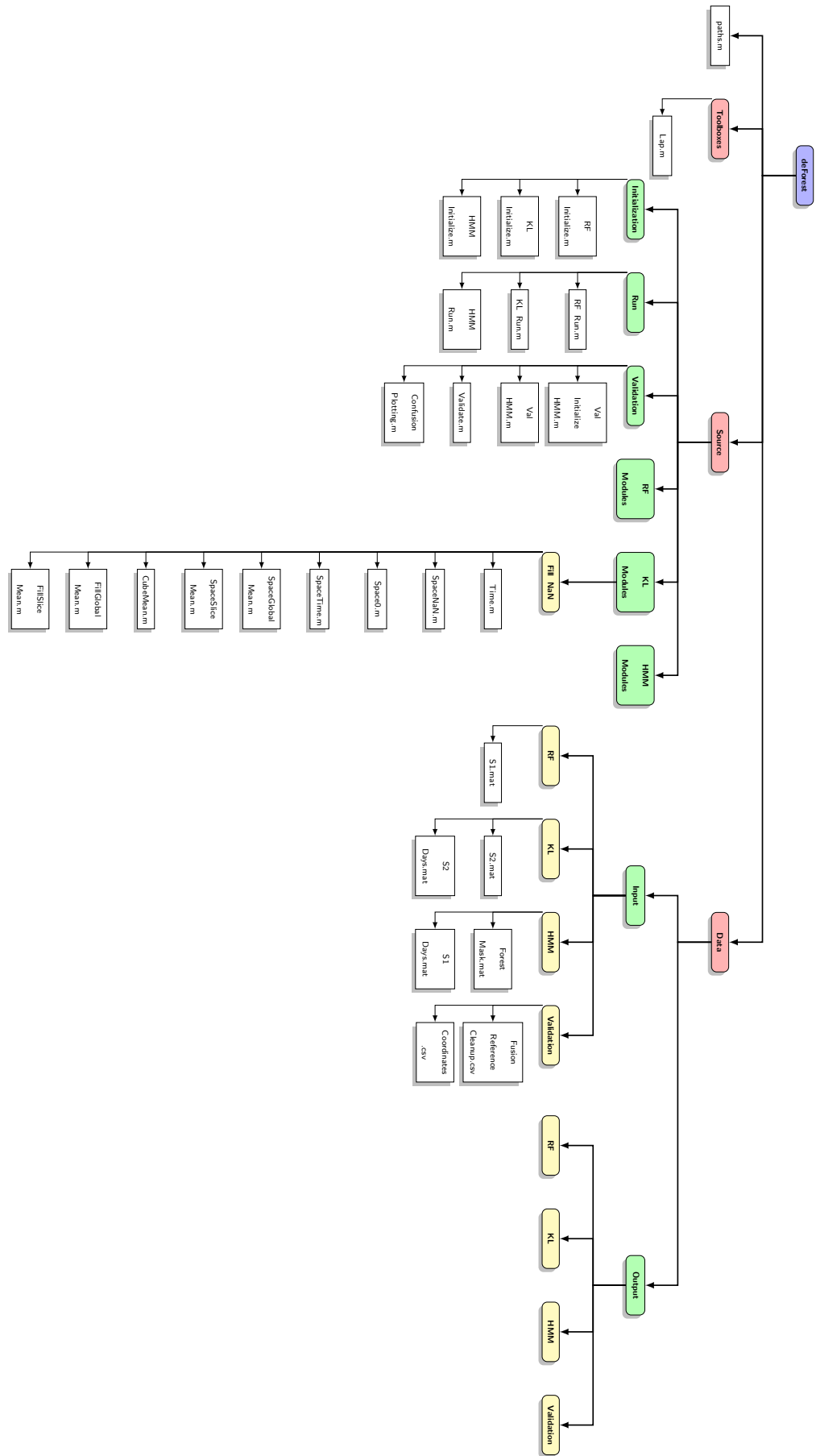


Figure 1: File structure for the deForest code.

Table 2: Initialization parameters for **KL**

Parameter Name	Parameter Description
MinFilter	Lower bound on EVI Values. Anything below is set to NaN
MaxFilter	Upper bound on EVI Values. Anything above is set to NaN
numEigen	Number of eigenvalues to use in the truncated KL expansion.
Maxtime	Number of days used for training.
fill_flag	Flag for whether to fill missing data (1) or leave missing data as NaN (0).

By default fill_flag is set to 0, which means that missing data is left as missing. This is how the data was processed for the results in the paper. However, Supplemental Section S.V shows a modest performance increase from using some of the data fill files in `/Source/KL_Modules/Fill_NaN/`, with the empirical best choice being Space_0 with $k = 3$. All files that start with Space fill using the mean of k -nearest neighbors in a region centered at the pixels that extends up/down in time by 1 pixel and out in space by s pixels. The difference between the Space files is what they fill with if there are no neighbors in that region. The details for these files are listed in Table 3. The file used can be set in the **KL** Run file.

Table 3: Optional **KL** files for filling missing optical training data.

File Name	File Description
Time.m	Fill using the mean of k -nearest neighbors to a pixel in time.
Space_NaN.m	Fill with NaN if there are no neighbors.
Space_0.m	Fill with 0 if there are no neighbors.
Space_Time	Fill using the mean of t -nearest neighbors in time if there are no standard neighbors.
Space_Global_Mean.m	Fill with the mean of all radar data if there are no neighbors.
Space_Slice_Mean.m	Fill with the mean of all radar data in the pixels time slice if there are no neighbors.
Cube_Mean.m	Fill with the mean in a cube with side length $2k + 1$ centered at the pixel.
Fill_Global_Mean.m	Fill all missing data with the mean of all radar data.
Fill_Slice_Mean.m	Fill missing data with the mean of the radar data in its time slice.

3.3 HMM Initialization

HMM_Initialize.m requires the file paths for the Forestmask (the forest/deforest classification for the initial state of the region), as well as the anomaly data, filtered radar data, list of radar days, and the output file path. The anomaly data and filtered radar data are the **KL** and **RF** outputs respectively. The Forestmask and radar day files are `/Data/Input/HMM/Forestmask.mat` and `/Data/Input/HMM/S1_Days.mat`. The **HMM** parameters are listed in Table 4.

Table 4: Initialization parameters for **HMM**

Parameter Name	Parameter Description
flag	Which method is run (1 is Optical Only, 2 is Radar Only, 3 is Hybrid)
radar_index	Channel number in the radar data that contains the filtered radar data.
anomaly_threshold	Threshold for mapping the anomaly values to binary
radar_threshold	Threshold for mapping the filtered radar values to binary
n_frames_to_confirm_class	Frames to confirm (FTC) in postprocessing
subregion_height/width	Dimensions of the subregions that the data processed in. RAM/Runtime tradeoff.
Maxtime	Number of days used for training.
number_of_workers	Number of threads to run on. Multithreading substantially decreases runtime.

The data is processed in chunks with size set by subregion_height/width, and the code only loads into memory the data needed for the current chunk. This lowers the RAM requirement substantially at the cost of a minor runtime increase.

4 Running The Code

1. Download the input data from Dropbox (https://www.dropbox.com/scl/fo/necxucp50n8yd7jr1u0zg/APiyR_ScRBNGnYP-b5QHRtA?rlkey=xh706o5gi44gxic5ajlwxxz97m&e=1&st=x5mj36k8&dl=0) and put the files in the sub-folders listed in Figure 1
2. Run *paths.m* to set the directory (this file will need to be localized first)
3. Set the folder paths, file paths, and parameters in the **/Source/Initialization/** files.
4. Run **/Source/Run/RF_Run.m** to generate the filtered radar data. Using 28 cores this requires about 150 GB of RAM and 22 hours.
5. Run **/Source/Run/KL_Run.m** to generate the anomaly data. The optional NaNFill method is specified in this file. Using 28 cores this requires about 390 GB of RAM and 3.5 hours.
6. Run **/Source/Run/HMM_Run.m** to generate the deforestation bitmap and datemap. Using 28 cores the hybrid method requires about 50 GB of RAM and 40 hours. Optical only and radar only require roughly half the RAM and runtime. These values are also highly dependent on specified subregion size.

Note: Code run using two 14 core 2.4 Ghz Intel Xeon E5-2680v4 CPUs

5 Validation

/Source/Validation/ contains the files needed to create the random removal figures in the paper.

- The main file is **/Source/Validation/validate.m**, which cuts the anomaly, radar, and forestmask data for the validation pixels out of the files for the large area, then randomly selects subsets of the optical data and feeds these new files to **/Source/Validation/val_initialize_hmm.m**.
- **/Source/Validation/val_initialize_hmm.m** initializes **/Source/Validation/val_hmm.m**, which runs the HMM on these small subsets of the total data and passes the results back to **/Source/Validation/validate.m**,
- **/Source/Validation/validate.m** uses the hand classified validation data to create the confusion matrices associated with the HMM outputs.
- The confusion matrices for all the HMM outputs are saved in **/Data/Output/Validation/**, and this is used by **/Source/Validation/confusion_plotting.m** to calculate the various metrics and create the figures, which are then saved in **/Data/Output/Validation/**.

/Source/Validation/validate.m requires the file paths for the Validation input and output folders, the **HMM** input folder, the **RF** output folder, and the **KL** output folder. It also requires setting the `number_of_workers` parameter, which is the number of cores/threads to use, and possibly `K`, which is the number of times to run the HMM for a given number of optical days. In the paper this is set to 100.

/Source/Validation/val_initialize_hmm.m requires the file path for the Validation input folder, as well as the `radar_index`, `number_of_workers`, and `Maxtime` parameters (see Section 3.3).

/Source/Validation/confusion_plotting.m requires the file paths for the Validation input and output folders, as well as the `K` parameter.