LoadDef Frequently Asked Questions
Hilary R. Martens

1. **What is the main difference between the “classic” station-centered mode and the “new” common-mesh mode for the convolution?**

In "**classic mode**" – which uses station-centered grids – LoadDef creates a separate surface grid for each station or observation point. For example, if you are analyzing 1000 observation points, it will generate 1000 unique surface grids. Each grid has a central pole at the corresponding observation point, with grid cells radiating outward from this pole, similar to a geographic coordinate system with the observation point at the north pole. These grids determine the resolution and shape of the cells where the load and the load Green's functions are interpolated.

Sation-centered grids are beneficial when the observation points are close to the load because they offer finer resolution in the immediate vicinity. However, this approach can be computationally expensive, especially with many observation points, because each grid is unique and does not align with the others. Furthermore, if the resolution is coarse where the load is, it can result in noisy patterns in the results. Using the default grid resolutions, this would be especially noticeable when the observation points are far from the load. This misalignment might explain the issues you are observing, where different grids result in varying representations of the load, particularly when the observation points are distant from Greenland. The refinement near each observation point is high but decreases with distance.

In "**common-mesh mode**" – which you can set up with the "global mesh" function – LoadDef uses a single, unified surface grid for all stations or observation points. The grid can be customized, but the examples from the GRDGEN folder create regularly spaced geographic grids with added refinement in specific areas. You can adjust these settings as needed.

While the load Green's functions must still be computed and interpolated onto this common surface grid for each observation point (since the Green's function values depend on the distance between the observation point and the grid cells), the load itself can be interpolated onto the grid beforehand. Because all observation points use the same grid, there is consistency in how the load is represented, and the computational time is reduced.

However, the common mesh approach has its downsides: you cannot achieve high refinement near every station unless they are very close to each other (not globally distributed). A globally refined mesh would be computationally expensive.

2. **In the sensitivity paper (Martens et al., JGR, 2016) and the Tutorial_May2023.pdf, the horizontal Load Green's Functions display negative values within a 10-degree radius from the load point, with negative values indicating the observer**

moves towards the load point (shortening the horizontal distance between them). However, Example 2a in Tutorial_May2023.pdf shows positive lateral displacements, which seems contradictory. In one example, the results are multiplied by -1 before plotting, so the sign was flipped. Could you help to explain this apparent discrepancy?

The direction of positive lateral displacement depends on the definition of the spherical coordinate system used. For the load Green's function with a delta-function point load, we typically place the load at the top of the coordinate system (positive "z" direction). In this setup, the "theta" vector points positively outward from the load point. Consequently, because positive "theta" is defined as outward, lateral displacements near the load are negative, indicating they are directed inward towards the load. Thus, applying a load results in lateral shortening (as you correctly deduced).

For spatially distributed loads on Earth, such as ocean tides, LoadDef converts the coordinates to east and north geographical directions to eliminate ambiguity. In this system, lateral displacements are positive in the east and north directions (and negative in the west and south directions).

In the disk example presented in the tutorial, only the "analytical approach" involves multiplying the lateral displacement by "-1" before plotting. This adjustment ensures consistency with the "convolution approach" disk example, which follows in the tutorial. In the convolution approach, the disk load is centered at the geographical north pole. Consequently, lateral displacements are zero in the east component (due to the axisymmetric nature of the load) and positive in the north component (directed towards the load center). In contrast, the "analytical approach" does not consider geographical positioning directly. Therefore, multiplying by "-1" adjusts the displacement values to simulate the disk being placed at the geographical north pole as well.

3. **Is it possible to compute depth-dependent load Love numbers and load Green's functions for PREM?**

Yes, this is now possible. You will first need to compute the load Love numbers for PREM (or any 1-D, radially varying model of your choosing) at the desired depths beneath the surface. This is a relatively new feature within LoadDef. It is provided as an option within the "run_ln" program in the "working" directory. You will then compute the load Green's functions from your custom set of Love numbers.

Given that the feature is relatively new, I welcome your feedback, especially if you encounter anything unexpected. I consider this feature to be in the testing phase.

4. **Is it possible to use LoadDef to estimate terrestrial water storage (TWS) from GPS observations of Earth-surface displacement?**

Yes, this now possible. You first need to generate a grid of cells for your study region (GRDGEN folder). The grid of cells defines your model parameters – the locations and areas within which you will solve for TWS. You can optionally create a common mesh as well (GRDGEN folder) for your study region.

After you have created the grid of load cells (your model parameters) and optionally a common mesh (defining the resolution of the convolution and Green's-function interpolation), you will then create a design matrix. The design matrix contains the physics of the problem (i.e., how water mass deforms the Earth). To create the design matrix, you will navigate to the "desmat" folder, modify any of the user inputs in "run_dm_load" to suit your needs, and then execute "run_dm_load."

After your design matrix has been computed, you are ready to solve the linear system:
G m = d.
G is the design matrix, m is your model vector (i.e., load height in meters within each grid cell), and d is your data (i.e., observations of surface displacements associated with the water loading, such as from GPS stations).

Many tools are available to solve the linear system, including within NumPy's Linear Algebra library. If you like to discuss this further, don't hesitate to reach out.