PHD filtering recursion

Task: complete the code of the PHD filter class, which contains necessary functions to implement a Gaussian-mixture Probability Hypothesis Density (GM-PHD) filter:

1. PHD filter prediction;
2. PHD filter update;
3. extract object state estimates from (Gaussian mixture) Poisson intensity.

For the first task (PHD filter prediction), your implementation should consist of the following steps:

1. Predict each Gaussian component in the Poisson intensity for pre-existing objects.
2. Add (Gaussian mixture) Poisson birth intensity to (Gaussian mixture) Poisson intensity for pre-existing objects.

For the second task (PHD filter update), your implementation should consist of the following steps:

1. Construct update components resulted from missed detection.
2. Perform ellipsoidal gating for each Gaussian component in the Poisson intensity.
3. Construct update components resulted from object detections that are inside the gates.

There are a few alternatives when extracting object states from a Gaussian mixture multi-object density. In the third task (extract object state estimates), your implementation should consist of the following steps:

1. Get a mean estimate of the cardinality of objects by taking the summation of the weights of the Gaussian components (rounded to the nearest integer), denoted as n.
2. Extract n object states from the means of the n Gaussian components with the highest weights.

Note:

1. When calculating the weight of the new Gaussian component resulting from object detection, you should scale it properly by taking the clutter intensity into account.
2. In the estimator, you can use `n = min(n, #Gaussian components)` to make sure that your implementation still works if `n > #Gaussian components`.

Files referenced:

- *`log_mvnpdf.m`*
- *`normalizeLogWeights.m`*
- *`motionmodel.m`*
- *`hypothesisReduction.m`*
- *`GaussianDensity.m`*
- *`measmodel.m`*
- *`modelgen.m`*

Note that `obj.density` is a function handle bound to MATLAB class `GaussianDensity`. For instance, you can simply call `obj.density.update` to perform a Kalman update instead of using your own code.