Comps

Cari Comnick, Logan Crowl, Sophie Gunn, Aidan Mullan

April 3, 2018

You'll need to generally define scagnostics before launching into them

1 Introduction

I know you don't have intro materials yet, but start thinking of ways to intro scagnostics, and transitions that help flow from section 2 to 3.

2 Scagnostic Ideas

Geometric graphs

Add an example to anchor discussion.

Geometric graphs are used to compute the feature measures used in scagnostics. A graph is a set of vertices V which are related by edges $e(v, w) \in E$, with $v, w \in V$. For scagnostics, only geometric graphs are used. Geometric graphs are those that can be represented in a metric space S as points and lines. Additionally, scagnostics only use geometric graphs that are undirected (all pairs (v, w) are unordered), simple ($v \neq w$), planar (can be represented in 2-dimensional space with no crossed edges), straight (all edges are straight lines), and finite (V and E are finite).

Feature measures

Certain feature measures of geometric graphs are used to calculate scagnostics:

1. *Length*(e) is the Euclidean distance between the vertices of an edge e.
2. *Length*(G) is the total length of all edges of a graph G.
3. A *path* is a list of vertices such that all successive pairs are an edge.
4. A path is *closed* if its first and last vertices are the same.
5. A *polygon* is the boundary of a closed path.
6. *Area*(P) is the area of polygon P.
7. *Perimeter*(P) is the length of the boundary of polygon P.

Convex Hull

Clearer to define this first, then convex hull

A convex hull of a given set of points X is the intersection of all convex sets of X, thus making it the smallest convex set containing X. A convex set of X contains all the straight line segments containing any pair of points in X. In Euclidean space, the convex hull can be thought of as the polygon created by wrapping the set of points X in a rubber band.

Nonconvex Hull/Alpha Hull

There are many ways to represent the non-convex shape of a set of points. For scagnostic measures, the *alpha hull* is used due to its computational efficiency and status as an erosion method. The *alpha hull* is a graph where points are connected by an edge when they can be touched by an open disk $D(\alpha)$ containing no points. The value α represents the radius of the disk and is chosen to be the value of the ω parameter. The ω parameter is the cutoff value for identifying outlying edges in the minimum spanning tree (defined below). This choice for α will erase edges in the Delaunay triangulation that are longer than outlying edges in the original minimum spanning tree [maybe we should cut this sentence since we don't actually define Delaunay triangulation]. The *alpha hull* can be constructed by rolling a circle of radius α around the set of points and placing edges between points that touch the circle. It can also be generalized to n dimensional data sets by replacing the 2 dimensional open disk by an n dimensional open ball.

I agree. You need to mention Delaunay Δ to keep.

MST

Spell out

A *tree* is any simple graph that is undirected, connected, and acyclic. For a given set of points X, a *spanning tree* is any tree whose vertices are exactly the points in X. The *minimum spanning tree* of X is defined as the spanning tree whose total edge weight is the minimum for all possible spanning trees of X. In the context of geometric minimum spanning trees, edge weight is taken to be the Euclidean distance between the two vertices connected by a given edge.

3 Scagnostics

Clumpy

Be more specific. Not commonly used stat

Add citation

The clumpy scagnostic indicates the clustering of points. This measure uses the Hartigam and Mohanty RUNT statistic. [Using the single-linkage hierarchical clustering tree called a dendrogram,] wordy it uses the runt size of each node (r_j) - the smaller of the number of leaves of each of the two subtrees joined at that node. Each r_j is associated with an e_j, and the runt graph R_j corresponding to each edge e_j is the smaller of the two subsets of edges that are still connected to each of the two vertices in e_j after deleting edges in the MST with lengths less than $length(e_j)$. This measure emphasizes clusters with small intracluster difference relative to their connecting edge. In this formula, j indexes edges in the MST and k indexes edges in each runt set derived from an edge indexed by j.

Typographic comment: use \rm for function names

function?

$$c_{clumpy} = \max_j \left[1 - \max_k[length(e_k)]/length(e_j)\right]$$

[Here and w/ other measures]

Sparse

Sparseness measures whether points are conformed to a small number of locations on the plane. confined? This ~~can~~ happens with small numbers of points, ~~or if the plot is of~~ and when plotting categorical variables. It is measured using the 90th percentile of the edge lengths of the MST. If this exceeds 1, it is capped at 1.

$$c_{sparse} = q_{90}$$

Striated

Linking to example plots will help here.

The striated measure captures how smooth paths in the minimum spanning tree are. An example of this smoothness would be found in a plot of categorical versus continuous variables (producing stripes), but it can also be found in time series, curves, and smooth algebraic functions. To generalize, the measure is based on the number of adjacent edges (the set of adjacent edges is $V^{(2)}$) whose cosine is less than -0.75.

$$c_{striated} = \frac{1}{|V|} \sum_{v \in V^{(2)}} I(cos\theta_{e(v,a)e(v,b)} < -.75)$$

Convex

Convex: The convex scagnostic is the ratio of the area of the alpha hull to the area of the convex hull.

$$c_{convex} = area(A)/area(H)$$

Skinny

The skinny scagnostic is a normalized measure of the ratio of the area to the perimeter of the alpha hull. Normalization ensures a circle earns a value of 0, a square of 0.012 and a skinny polygon near 1.

clarify - missing words?

$$c_{skinny} = 1 - \sqrt{4\pi area(A)}/perimeter(A)$$

Stringy

A *stringy* plot is a skinny plot with no branches. Consider the diameter of a plot to be the longest shortest path through the minimum spanning tree. If the diameter is close to as long the total edge length of the MST then the plot has very few branches and should be considered stringy.

reword, clarify

$$c_{stringy} = \frac{\text{diameter}(T)}{\text{length}(T)}$$

What is T? Be sure to define everything

Monotonic

In order to measure the monotonicity of a set of points, the squared Spearman correlation coefficient is used. The Spearman correlation coefficient is the Pearson correlation on the ranks of the x and y coordinates. As a result, the Spearman correlation will be high when points have similar rank for the x and y coordinate and close to -1 when the coordinates have very dissimilar rank. In order to capture both positive and negative monotonic behavior, we take the square of this correlation as our measure of monotonicity.

$$c_{monotonic} = r_{spearman}{}^2$$

Outlying

check for grammar

The outlying scagnostic is a measure of the proportion of overall edge length that is due to the presence of long edges connected leafs, or points of single degree, in the minimum spanning tree. Edges are defined to be outliers if the edge weight is greater than ω, where ω is calculated by

$$\omega = q_{75} + 1.5(q_{75} - q_{25})$$

Here, q_{75} and q_{25} are the 75th and 25th percentile of edge lengths in the minimum spanning tree. Then, the outlying scagnostic is given as

Try to clarify a bit.

$$c_{outlying} = length(T_{outliers})/length(T)$$

Skewed

The skewness of a minimum spanning tree is a measure of the extreme-value distribution of the edge lengths. The skewed scagnostic is calculated by

Not quite true. We need a robust measure b/c of extreme values, but it doesn't mean the EV dsn.

$$c_{skew} = (q_{90} - q_{50})/(q_{90} - q_{10})$$

robust measure of skewness

4 Statistical Learning Algorithms

Logistic Regression

Logistic regression is used when the response variable is categorical and there are only two responses. The outcome can be predicted using one or more explanatory variables. In the case of our simulated training data, the dependent variable is signal (1 for signal, 0 for noise), and the explanatory variables are the nine scagnostics. In general, we model $p(X)$, the probability of the response $Y = 1$ given all explanatory variables X, is given by the logistic function

Not discussed yet.

Um... "For example, if detecting a signal plot from a field of nulls ..."

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}}$$

Additionally, the odds of the response $Y = 1$ can be written

$$\frac{p(X)}{1-p(X)} = e^{\beta_0+\beta_1 X}$$

A high odds indicates a high likelihood of being a success. If we take the log of both sides of the odds, we get a linear relationship between the log odds (logit) and X. The estimation of $\hat{\beta}_0$ and $\hat{\beta}_1$ is carried out by maximizing the likelihood function:

I find it clearer to discuss linear predictor $logit(\pi) = \beta_0+\beta_1 X+\ldots$ first, then give logit function. More "conventional"

Be direct.

New thought, new paragraph

$$l(\beta_0, \beta_1) = \prod_{i:y_i=1} p(x_i) \prod_{i':y_{i'}=0} (1 - p(x_{i'}))$$

For the final write up you might omit logistic regression since it's not used.

In our simulation cases, however, logistic regression did not work. Due to our data being simulated to measure specific scagnostics, we ended up with perfect separation in some models: the probability of a plot being signal was 1 ($p(X) = 1$). There is not a closed form for the maximum likelihood estimates of the $\hat{\beta}$s, but with perfect separation, cannot be calculated numerically as the likelihood approaches infinity and the estimates of $\hat{\beta}$s get very large. We can also see that the odds can't be calculated since it involves a division by 0; this makes sense as likelihood of a success approaches infinity if the probability of success is 1.

K-Nearest Neighbors

K-nearest neighbors is a classification method that classifies a new data point as being the same class as majority of its k nearest neighbors. We used Euclidean distance to determine the distances between points in our model, and tried k ranging from 1 to 25.

SVM

classification jargon

Could expand this by ~1 sentence + include some justification from logistic if omitted

We found support vector machines useful for our analysis when logistic regression did not perform well due to well separated data. A support vector machine is a non-probabilistic classifier with close ties to logistic regression. The main idea of SVM is that we want to find a boundary in the feature space of our data that divides the training data well. When this boundary is linear, this method is called the support vector classifier. In this case, the classifier determines a hyperplane that divides most of the training data correctly, but may misclassify some observations. A small amount of misclassification allows for greater robustness to individual observations as well as better classification of most training observations. However, sometimes a linear boundary does not separate the cases well, in this scenario we want to enlarge our feature space in order to allow for a non-linear boundary. Support vector machines allow us to do this using kernels, a computationally efficient approach to enlarge the feature space. The classifier is then defined as follows,

$$f(x) = \beta_0 + \sum_{i \in S} \alpha_i K(x, x_i)$$

Give example. R uses radial kernel by default. If you didn't Δ this, it's a good option.

Where K defines a kernel that is some measure of the similarity between x and x_i. β_0, $\alpha_1 ... \alpha_n$ are estimated using the inner products of the training observations.
This method gave us comparable results with LDA and QDA. more conclusion!

if appears before LDA/QDA section, spell out

Random Forest

Avoid mentioning what you won't do.

A random forest model is one of the more flexible statistical learning approaches because it can be used for quantitative or categorical response variables as well as any form of explanatory variable. Because classification is our primary objective, we will not touch on random forests in a regression context, but the algorithm naturally extends to quantitative response variables. At the base of every random forest is the decision tree algorithm. A decision tree produces a series of binary splits based on the predictor set in order to divide the data into homogeneous subsets. In order to determine a split, the algorithm considers all possible splits of each predictor and chooses the split that results in the largest possible drop in node impurity. Node impurity is typically measured using either the Gini impurity or total entropy–two numerically similar measures of total variance across all classes. This splitting procedure is then repeated for the two resulting subsets and the subset whose maximal split produces the biggest drop in impurity is split. This process repeats until a stopping criteria is met. The predicted class of an observation is generally the majority class of the terminal node in which it falls.

Focus on trees here only

Define measures you mention, or just the one used.

While decision trees offer an intuitive way of classifying observations, they are not very robust, which makes them prone to overfitting. ~~In order to combat this disadvantage, random forests use many decision trees and aggregate their results~~. In order to avoid overfitting, random forests begin by drawing a bootstrap sample to use as a training set. Then, a random subset of the overall predictor set is considered at each split in order to decorrelate the trees. Each tree is grown deep, so that it is low bias and high variance. This process is repeated for potentially hundreds of trees, and the accuracy of the model is determined by predicting an observation using the trees trained on bootstrap samples that did not include that observation. The final prediction is made by receiving a prediction from each individual tree and taking the majority prediction for that observation.

Nice transition

explain

LDA and QDA

Describe LDA/QDA first, then draw comparisons

Logistic regression can be effective when modeling data into two response classes that are not well separated. However, if we are concerned with data that can be sorted into more than two classes or classes that are well separated, linear discriminant analysis proves more robust. *Linear discriminant analysis* (LDA) allows us to model the probability that a given observation belongs to a particular response class indirectly by first modeling the distribution of predictors for each response class individually.
LDA assumes that ~~our~~ the data $X = (X_1, X_2, \ldots, X_p)$ are drawn from a multivariate normal distribution $X \sim N(\mu_k, \Sigma)$ where $\mu_k = E(X|Y = k)$ is the class-specific mean vector and $\Sigma = Cov(X)$ is a covariance matrix shared by all classes. If we denote K to be the set of all classes, our classification function for an observation $X = x$ is given by:

$$\delta(x) = \max_{k \in K}[x^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + \log \pi_k]$$

where π_k is the prior probability that the observation $X = x$ belongs to the k^{th} response class. Often this prior is taken to be the proportion of training observations that belong to the k^{th} class. It is important to note that the LDA classifier is trying to approximate the Bayesian classifier, which would give us the lowest total error rate out of all classifiers, assuming that our initial assumption of multivariate linear data holds. One key element of this assumption is that all classes follow the same covariance matrix. However, this may not always be the case.

Space →

We can relax this assumption of a common covariance matrix by using *quadratic discriminant analysis* (QDA). Now, we assume that observations belonging to a given class k are drawn from a multivariate normal distribution $X \sim N(\mu_k, \Sigma_k)$ where both the mean vector and covariance matrix are class-specific. For a given observation $X = x$, this gives us the classification function

$$\delta(x) = \max_{k \in K}[-\frac{1}{2}(x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k) - \frac{1}{2} \log |\Sigma_k| + \log \pi_k]$$

Space →

where π_k is the prior density for response class k.
The main distinction between LDA and QDA is in the bias-variance trade-off. LDA assumes a shared covariance across all classes, making it less flexible as a classifier. This in turn leads to a decreased variance as compared to QDA, but may suffer from increased bias should the assumption of a common covariance matrix be wrong. Both methods of classification were used during the analysis of our simulated data, which will be discussed later.

5 Simulation Methods

→ Overall, section is rather clear. Think about how to make a bit more concise. Also, add example plots for clarity.

We simulated bivariate 2D point distributions to validate the use of scagnostics in determining various plot features, and to ~~use~~ as the basis for our application's model and our lineups.

Striated

Bullets or table could help avoid "We ___. We ___"

For data to measure the striated scagnostic, we generated scatterplots consisting of "stripes" of points. We chose the number of points n in each stripe (50, 100, 150, 200, 250), the number of stripes s (2 to 8), and the distance between each stripe in a plot (1, 2 or 3). We also defined an error after we choose a distance d between stripes ($d/5, d/10, d/15, 0$ and d), used to jitter around each stripe. We also varied whether the stripes were horizontal or vertical. To set the locations of the stripes, we generate our first set of numbers. These were generated by replicating the sequence $a_n = dn, n \in [0, ..., d(s-1)]$. For example, if there were 3 stripes and distance 2 between them, then the numbers generated would be the set (0, 2, 4) repeated n times. Then, we jittered these numbers by adding a random uniform variable from $Unif(-error, error)$ to each number. To generate the other coordinates, we choose n times the number of stripes points from a $Unif(0, 1)$ distribution. Then we assign the two sets of numbers to be x or y coordinates based on the orientation. The plots with error equal to distance between stripes were the noise plots; we generated 840 signal plots and 210 null plots.

Show example of signal and example of noise.

Funnel

We used a bivariate log-normal to generate funnel plots. To generated the bivariate log-normal, we exponentiated a multivariate normal with ρ from .4 to .9 by .05, standard deviation from .3 to 1 by .05, and $\mu_1 = \mu_2 = 0$, so each plot was centered at $(0, 0)$. The number of points we used were 100, 200, 300, 400, and 500. We generated null plots in a pairwise fashion: for each signal plot, we generated another plot with everything the same, but with $\rho = 0$. We had 825 signal and 825 null plots.

Cluster

To generate plots used to illustrate the clumpy scagnostic, we generated scatterplots consisting of three clusters. We put the center of each cluster at three evenly spaced angles of a circle centered around the origin. We varied the radius of the circle unit-wise from 1 to 5. We then varied the starting angle - the location of the first cluster - around the circle at every 45 degrees. From the radius and starting angle, we chose three (x, y) coordinates to be centers: at the starting angle and then every 120 degrees so they are evenly spaced. Then we generated a cluster of points (n = 33, 67, 100, 133, 167 in a cluster; same n for the entire plot) around each point from a bivariate normal distribution with standard deviation (ranged from radius/8 to radius/3) and covariance 0. Since we set both standard deviations equal, each cluster was sampled from a circle. Each cluster signal plot had a corresponding null plot generated with the same n, radius, and three starting points. Each set of generated points was taken from a bivariate normal again, but this time with standard deviation r (still 0 covariance). We had 7200 signal plots and 7200 null plots.

Quadratic

Quadratic trends were modelled according to a quadratic equation with random noise. Off the 20,000 plots generated, half were signal and half noise. Sample sizes of 100, 200, 300, 400, and 400 were used with 2000 null and 2000 signal plots generated for each sample size. For each trial, values were sampled as follows:

$$a \sim Unif(-10, 10)$$

$$b \sim Unif(-10, 10)$$

$$\sigma \sim Unif(0, min(min(abs(a, b)), 1))$$

$$\epsilon \sim norm(0, \sigma)$$

Then, the x, y coordinates for the signal plots are generated as follows:

$$X_i \sim Unif(-1, 1)$$

$$Y_i \sim ax_i^2 + bx_i + \epsilon$$

Null plots were generated by sampling randomly from the X and Y values in the signal plots. These plots were then fed into the scagnostics r package and scagnostics values were saved as well as their signal/null identities.

Linear

Linear trends were modeled using the multivariate normal distribution with various sample sizes, and standard deviations. Of the 2500 different plots generated, 1250 signal plots and 1250 null plots. Of each type, sample sizes of 100, 200, 300, 400 and 500, n, and standard deviations of 1,2,3,4,5 for both the x and y directions were used (σ_1 and σ_2). For each unique combination of n, σ_1 and σ_2, 10 plots were created each with a unique correlation value r. r determines the strength and direction of the correlation. To generate appropriate r values a "cutoff" value was found using the t-distribution, considering the sample size, which determined whether the correlation would be considered significant or not. The reasoning behind this is that for some small n, we would not expect small correlation values to be detected. We then took this cutoff value and then sampled uniformly five values from -1 to –cutoff and five values from cutoff to 1. These r values were then used to generate our signal plots. For our null plots, ten r values were sampled uniformly from –cutoff to cutoff, and then these were used to generate our null plots. These plots were then fed into the scagnostics r package and scagnostics values were saved as well as their signal/null identities.

Donut

Plots were generated to form two concentric circles of varying width, or in more simple terms "donut" shaped plots. Of the 16,000 plots generated half were null and half were signal. Signal in this simulation indicates two separate donut shapes while null is only one donut shape. An equal number of plots with sample sizes of 300, 400, 500 and 600 were generated with each plot centered at $(0, 0)$ and of maximum radius one. For the signal plots the limits of the width of each donut was sampled from a uniform distribution. Each null plot corresponds to a signal plot where the width of the null plot's donut shape is the outer limits of the signal plots width (this does not make sense, do we even need it?).

These plots were then fed into the scagnostics ~~r~~ package and scagnostics values were saved as well as their signal/null identities.

QQ Plots

In order to generate quantile-quantile plots that display a variety of visual cues for non-normality, four different types of non-normal data were simulated with a collection of different parameters. Specifically, 16,000 QQ plots were created from randomly simulated data drawn from normal, log-normal, Chi-squared, exponential, and t distributions. Sample sizes of 20, 30, 40, and 50 were used for all distributions and t and Chi-squared distributions were simulated using 3, 4, 5, 6, and 7 degrees of freedom. For the other distributions, parameters were selected in order to create approximately equal variance for plots of the same sample size. Lastly, visual checks were performed to ensure that the non-normal QQ plots could in fact be discerned from the plots of the normal data.

Logistic Residials

Our primary concern for residual plots was recognizing problematic patterns within the residuals. As a result, we simulated our residual plots from data that violated the independence assumption for logistic regression. In order to create dependence between consecutive observations, we simulated a continuous predictor as a time series drawn from an autoregressive model with varying values of correlation between observations. We then used these predictor values to determine the probability of success for each individual observations (based on a linear function). Finally, we simulated our response variable using a binomial distribution using our correlated probabilities as the success probabilities for each trial. Using this correlated data, a logistic model was fit and a residual plot created. For the null plots, the same procedure was used, but a continuous explanatory variable was drawn from independent observations.

Time Series Data

Plots were generated from $ARMA(p, q)$ models where p and q were either 1 or 0. The magnitude of the AR and MA components was varied $(0.3, 0.5, 0.7, 0.9)$, and plots contained $100, 200, 300, 400$, or 500 points. Plots generated from a white noise model, ARIMA(0,0,0), were classified as "noise" and all other plots were classified as "signal." 3600 plots were simulated in total, with 900 plots created from each of ARMA(0,0), ARMA(1,0), ARMA(0,1), and ARMA(1,1) models.

Exponential

Coordinate pairs were constructed for the exponential simulation by randomly sampling x values from an exponential distribution with rate parameter $0.5, 0.75, 1, 1.5$, and 2. For each plot, between 100 and 500 values were drawn, similar to the other simulations. The corresponding y value for each pair was calculated by taking $e^x + \epsilon$, where ϵ was drawn from a normal distribution with mean 0 and a standard deviation of $\frac{e^x}{s}$ with s being one of $1, 2, 3, 4, 6, 8, 12$. Every exponential plot was paired with a "noise" plot constructed by randomly sampling x and y values from normal distributions with mean and standard deviations calculated from the x and y values of the exponential simulation. 3500 plots were generated in total, with 1750 being "noise" and 1750 being "signal."

Linear Residual

Initial scatter plots were constructed by randomly sampling coeficients a and b, as well as all x values from a uniform distribution between -10 and 10. Between 100 and 500 x values were drawn following similar procedure to other simulations. For each random drawing, three sets of y values were constructed. First, a linear set of y values was created according to $y = ax + b + \epsilon$ were ϵ was a random error term drawn from a normal distribution with mean 0 and standard deviation randomly chosen from a uniform distribution from 0 to 10. Next, a quadratic set of y values was drawn following $y = ax^2 + bx + \epsilon$ using the same ϵ as before. Lastly, a non-constant variance set of y values was created according to $y = ax + b + \eta$ where η is drawn from a normal distribution with mean 0 and standard deviation $\sqrt{x^2}$. Each of these three sets of points was fit with a linear model, and residuals were calculated. These residual plots were the plots of interest, with x values being the fitted values for the models and y values being the model residuals. Residual plots generated from the linear y sets were labeled as "noise" and the quadratic and non-constant variance residuals were labeled as "signal." 7500 total plots were created, with 2500 "noise" and 5000 "signal."

A few other thoughts:

- Start including citations. I recommend using Bibtex to manage them.
  - → Don't forget to cite the scagnostics package. citation(package = "scagnostics") will print the Bibtex entry
- Focus on what you did, no need to mention what you didn't do until discussion / future directions.
- When editing, strive for concise sentences. Use active voice.
- Remember that the reader will know that you all are the authors, so you can avoid phrases like "In our opinion..."