Matrix Completion Techniques For Anomaly Detection in Network Attacks

---

A Thesis
Presented to
Department of Statistical Science
Duke University

James C. Wu

May 2018

Approved for the
Bachelor of Science in Statistical Science

Peter D. Hoff

Jerry P. Reiter

Galen Reeves

Mine Cetinkaya-Rundel, DUS

Acknowledgements

I thank my advisor, Professor Peter Hoff, and the Director of Undergraduate Studies, Professor Mine Cetinkaya-Rundel, for their guidance in this project. I also thank Duke University's Statistics Department for supervising this project and the Office of Information Technology for providing the dataset. Most of all I thank my parents for their continued unwavering support in all my endeavors.

Table of Contents

Abstract

The goal of this project is to identify novel methods for detecting anomalies in network IP data. The space is represented as a 3-dimensional tensor of the continuous features (source bytes, destination bytes, source packets, destination packets) divided by their respective source port and destination port combinations. This project implements and assesses the validity of principal component analysis and matrix completion via singular value decomposition (more methods pending) in determining anomalous entries in the tensor.

The goal of this project is to identify novel methods for detecting anomalies in network IP data. The space is represented as a 3-dimensional tensor of the continuous features (source bytes, destination bytes, source packets, destination packets) divided by their respective source port and destination port combinations. This project implements and assesses the validity of principal component analysis and matrix completion via singular value decomposition (more methods pending) in determining anomalous entries in the tensor.

Chapter 1

Introduction

1.1 Anomaly Detection

Anomaly detection is used to identify unusual patterns or observations that do not conform to expected behavior in a dataset. Anomalies can be broadly categorized into three categories:

Point anomalies: A single instance of data is anomalous if it's too far off from the rest. For example detecting credit card fraud based on a single spending spree that represents the credit card being stolen and used.

Contextual anomalies: The abnormality is context specific. This type of anomaly is common in time-series data. For instance, high spending on food and gifts every day during the holiday season is normal, but may be considered unusual otherwise.

Collective anomalies: A set of data observations that when collectively assessed helps in detecting anomalies. For instance, repeated pings from a certain IP address to a port connection on a hosted network may be classified as a port scanner, which often preludes a network attack.

1.2 Network Attacks

Network security is becoming increasingly relevant as the flow of data, bandwith of transactions, and user dependency on hosted networks increase. As entire networks grow in nodes and complexity, attackers gain easier entry points of access to the network. The most benign of attackers attempt to shutdown networks (e.g. causing a website to shutdown with repeated pings to its server), while more malicious attempts involve hijacking the server to publish the attacker's own content or stealing unsecured data from the server, thus compromising the privacy of the network's users.

Attackers follow a specific three step strategy when gathering intelligence on a network, the most important component of which is scanning. Network scanning is a procedure for identifying active hosts on a network, the attacker uses it to find information about the specific IP addresses that can be accessed over the Internet, their target's operating systems, system architecture, and the services running on each node/computer in the network. Scanning procedures, such as ping sweeps and port scans, return information about which IP addresses map to live hosts that are active on the Internet and what services they offer. Another scanning method, inverse mapping, returns information about what IP addresses do not map to live hosts; this enables an attacker to make assumptions about viable addresses.

All three of these scanning methods leave digital signatures in the networks they evaluate because they apply specific pings that are then stored in the network logs. Most scanners use a specific combination of bytes, packets, flags (in TCP protocol), and ports in a sequence of pings to a network. Identifying a scanner's often many IP addresses from the set of pings available in the network's logs is thus an anomaly detection problem. In particular, because the data is unlabeled, meaning it is unclear which observations are actually scanners and which are just standard user behavior, unsupervised approaches are necessary for tackling the problem.

This particular dataset is from Duke University's Office of Information Technology (OIT), and it covers all observations in their network traffic during a five minute period in February 2017.

1.2.1 Status Quo Solution

OIT's current solution for detecting scanners relies on specific domain knowledge gathered from diagnostics programs and data analysis completed on previous data. They prevent scanners by blocking IP addresses that fit certain rules they have constructed to run on every network transaction as it occurs. The specific checks in these rules are private for security reasons, but they belong to the nature of evaluating the size of transactions, repeated connections between particular ports, many pings from the same address, and combinations of these particular behaviors.

While this solution presents a methodical way for banning IP addresses and its method of rule checking is essentially removing what OIT considers outliers for network transactions-any observation that does not fit within the constraints specified by the rules is classified as an outleir and its source IP is blocked-it is inflexible, prone to detecting false negatives, and fails to detect observations that may be within the parameter constraints of the rules but are anomalous with respect to other parameters or parameter constraints.

1.3 Network Dataset

1.3.1 Features

The networks dataset contains 13 features, 8 categorical and 5 continuous, and the observations are unlabeled (not specified whether they are considered a scanner). The 13 features are:

Continuous:

- StartTime (Start Time): the time when the observation is logged
- SrcBytes (Source Bytes): the total number of bytes sent in the observation
- SrcPkts (Source Packets): the number of packets sent in the observation
- DstBytes (Destination Bytes): the total number of bytes received in the observation
- DstPkts (Destination Packets): the number of packets received in the observation Note, the destination packets and bytes features do not have the same values as their source counterparts because the connections are compressed and decompressed into different forms and byte sizes when sent. For instance, it is possible for the number of destination packets to be larger than source packets. It is also possible for information to be lost during the connection.

Categorical:

- Flgs (connection flag): flow state flags seen in transaction between the two addresses
- Proto (network protocol): specifies the rules used for information exchange via network addresses. Transmission Control Protocol (TCP) uses a set of rules to exchange messages with other Internet points at the information packet level, and Internet Protocol (IP) uses a set of rules to send and receive messages at the Internet address level.
- SrcAddr (Source Address): the IP address of the connection's source
- DstAddr (Destination Address): the IP address of the connection's destination
- Sport (Source Port): the network port number of the connection's source. A port numbers identifies the specific process to which a network message is forwarded when it arrives at a server.
- Dport (Destination Port): the network port number of the connection's destination
- Dir (direction): the direction of the connection
- State (connection state): a categorical assessment of the current phase in the transaction when the timestamp is recorded

Note, the addresses have been anonymized for security reasons.

1.3.2 Argus

Argus is the open source network security tool applied to network transactions that collects the data for the features. The Argus wiki and the OIT manual provides key insights into the structure and nature of the data. Specifically, the sessions are clustered together by address, so the pytes and packets values are accumulative over a set duration and each session has its own start time but does not have a tracked end time. There exist 2-4 million connections on average every 5 minutes. Furthermore the protocol in this dataset is always gathered from TCP protocol and the direction will always be to the right (i.e. Source to Destination). This information supports dropping proto, StartTime, and Direction from the dataset for future analysis because they do not present any information regarding whether an observation can be considered an anomaly. Furthermore, the State feature may not be reliable because Argus occasionally resets the state data statistics during monitoring.

Chapter 2

Modeling Port Relationships

2.1 Motivation

Preliminary data analysis signaled that there may exist trends between different port combinations. For instance, a particular source and destination port may frequently contain large byte transactions in their connections. Devising a systematic way to identify these combinations may present outliers that can be further investigated for scanner behavior.

This approach to the anomaly detection problem reduces the dataset to the values of the four continuous features, SrcBytes, SrcPkts, DstBytes, DstPkts, observed across different source port and destination port combinations. The data can be represented as a 3-dimensional tensor $Y \in \mathbb{R}_{m \times n \times 4}$ where m represents the number of source ports, n represents the number of destination ports, and 4 accounts for the four continuous features in the dataset. Each cell, y_{ijk}, contains the mean of all the observations observed for source port, i, and destination port j. In the cases where the combination of i and j is not observed in the dataset, y_{ijk} is missing.

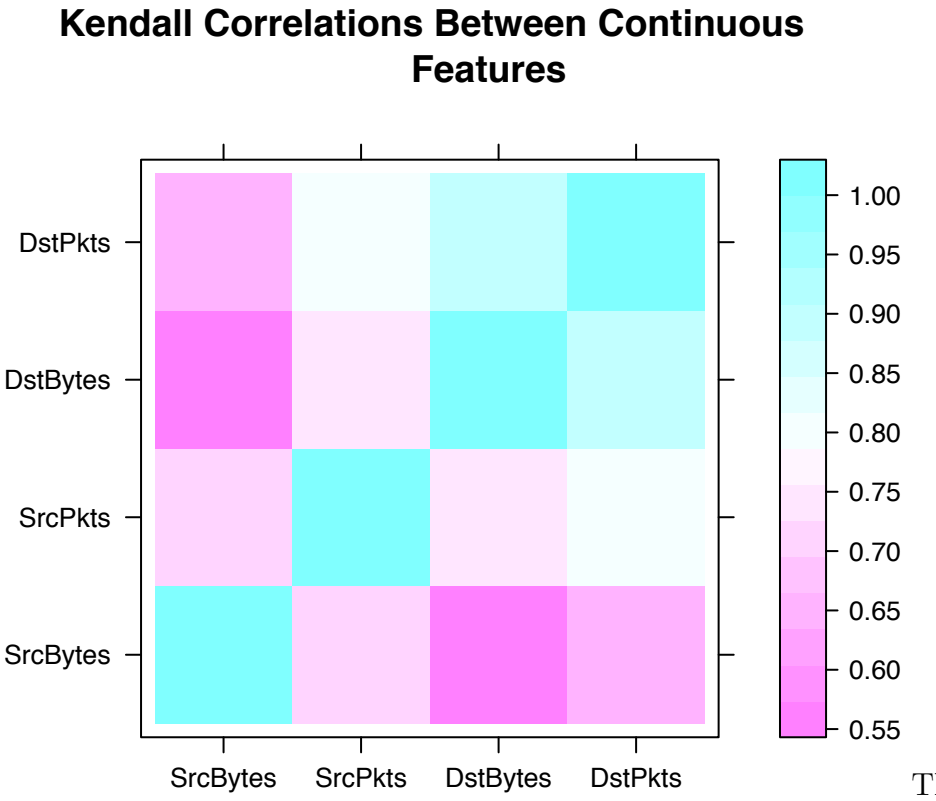
The goal of this paper is to devise an optimal strategy for imputing the missing cells in Y to create the completed tensor $Y' \in \mathbb{R}_{m \times n \times 4}$. As new observations are observed for combinations of ports i and j, the y'_{ijk} values can be interpreted as an approximation for the expected behavior for that particular port combination. Observations with continuous features that are a certain threshold away from y'_{ijk} may be marked as anomalies and investigated further.

Imputing values for each of the four continuous features in the dataset for all possible source and destination port combinations yields a reasonable expected value in each cell of the ports matrix that can then be compared to actual connection values when they are observed. New observations that differ greatly from the imputed values are flagged as anomalies and require further investigation.

2.2 Tensor Properties

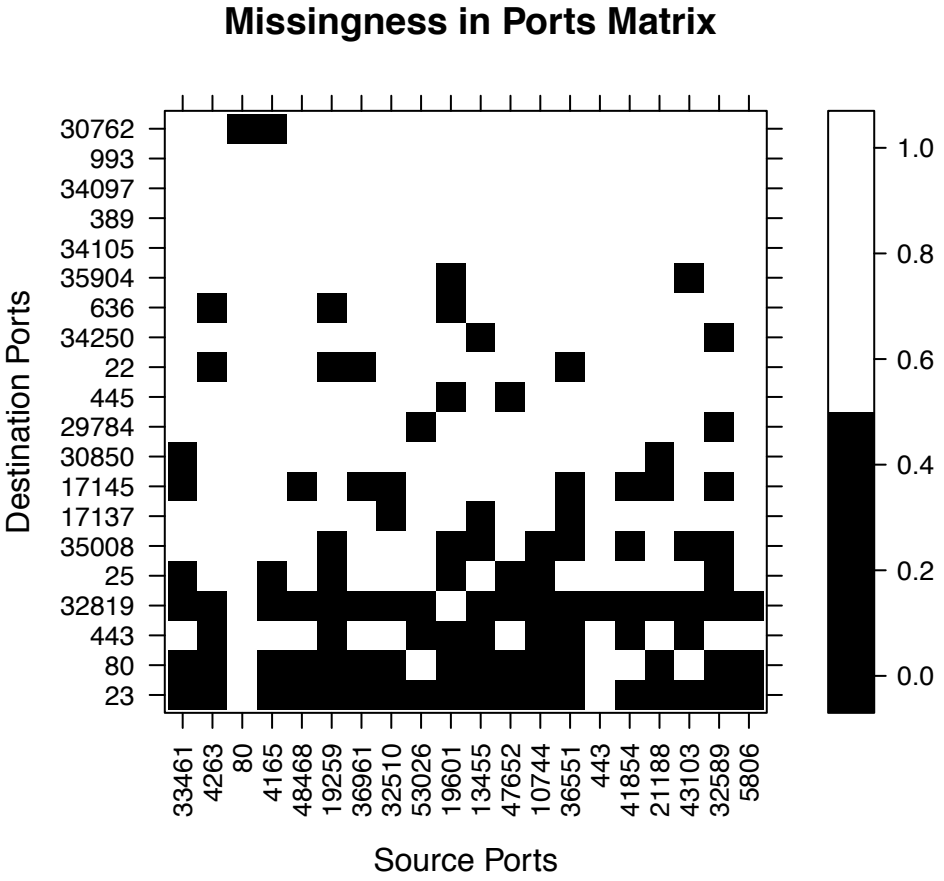
The following properties of Y inform the imputation strategies in following sections.

2.2.1 Correlations



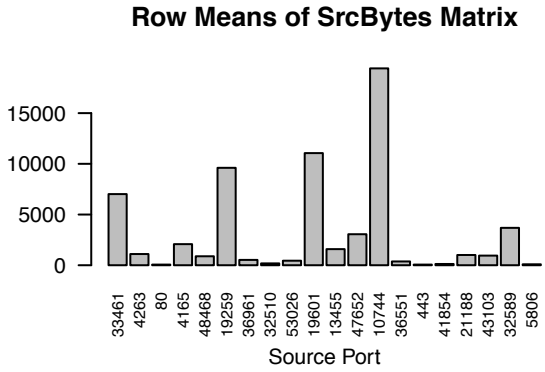
The matrix above describes the Kendall rank correlations (commonly referred to as Kendall's tau coefficent) between the four continuous features in the dataset. Intuitively, the Kendall correlation between two features will be high when observations have a similar rank (i.e. relative position label of observations within the variable: 1st, 2nd, 3rd, etc.) between the two variables, and low when observations have a dissimilar rank between the two variables. The range of correlations is [-1, 1]. Kendall correlation was selected as a measure because it evaluates ranks between observations, as opposed to Pearson, which is more susceptible to outliers in the dataset (large byte and packet observations in the continuous feautres skewed the Pearson measures).

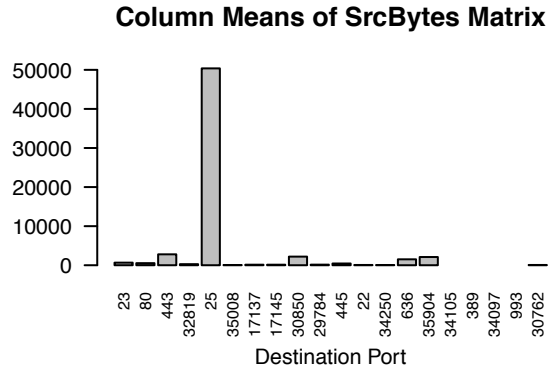
2.2.2 Missingness

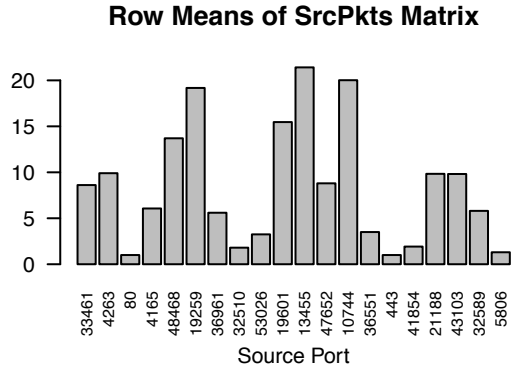


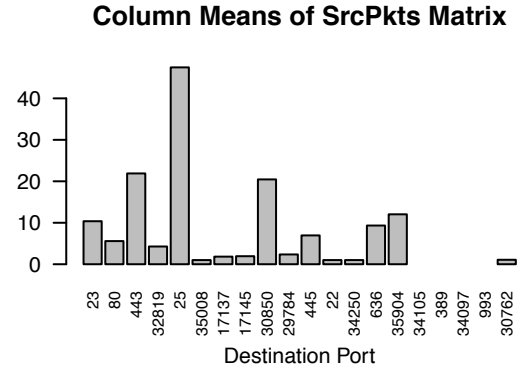
The above matrix represents the missingness in the port combinations for pairings of the top 20 most used source ports and destination ports. The black cells represent missingness; of the 400 cells in the matrix, 295 (73.75%) of cells are missing observations.

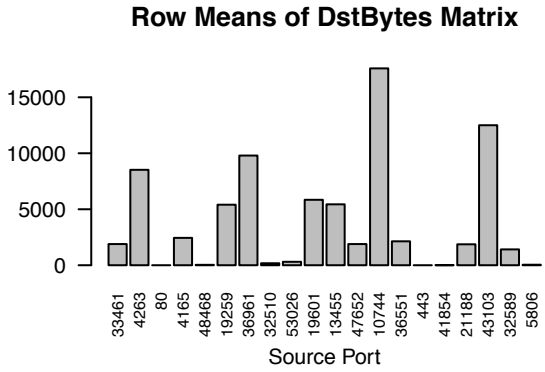
2.2.3 Row and Column Properties

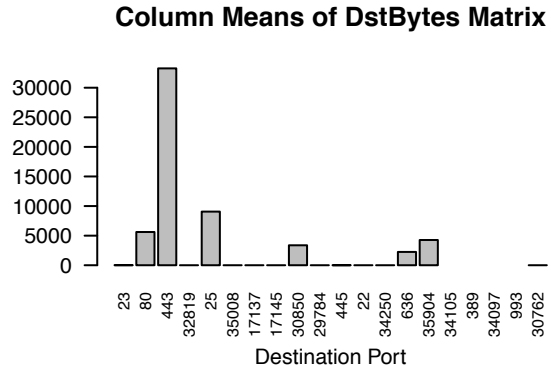


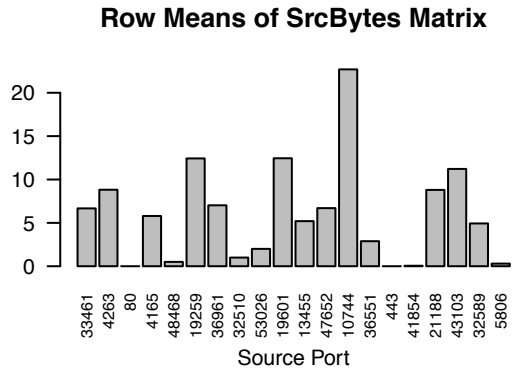


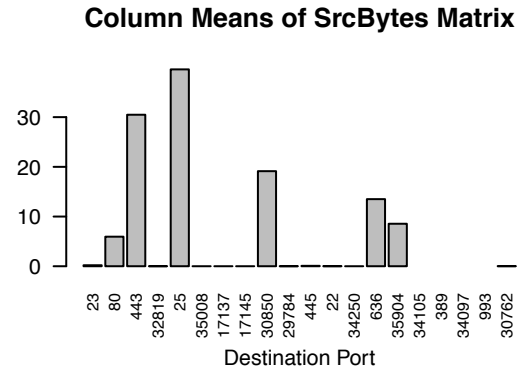






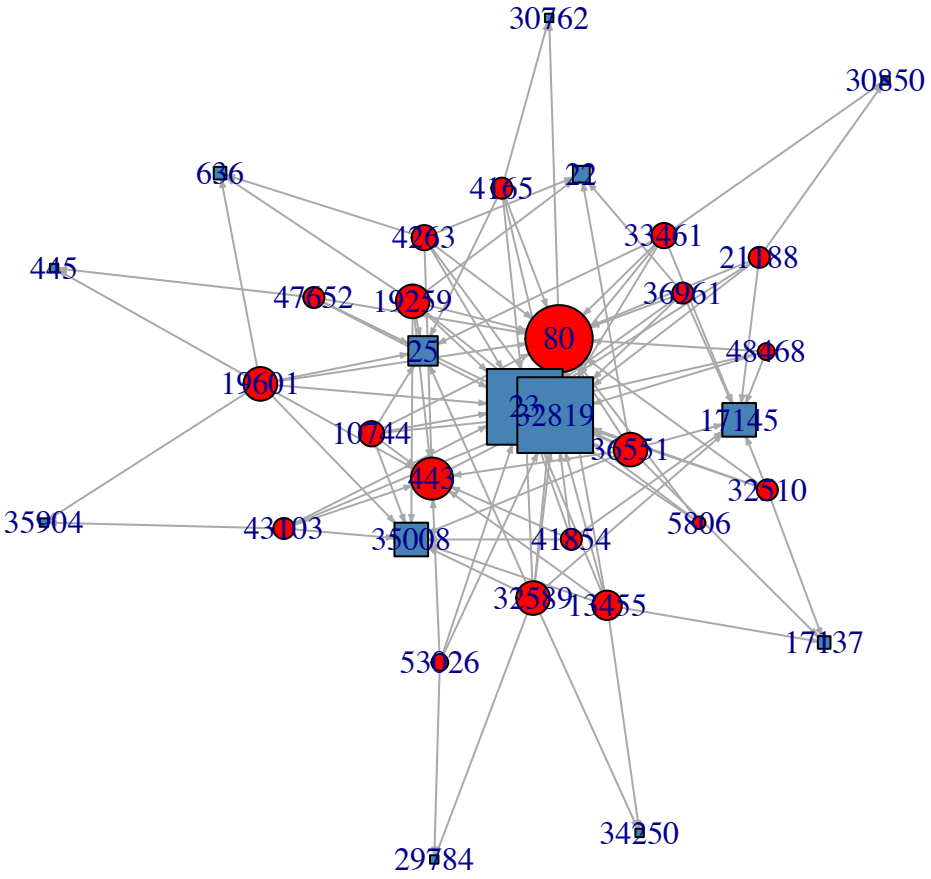






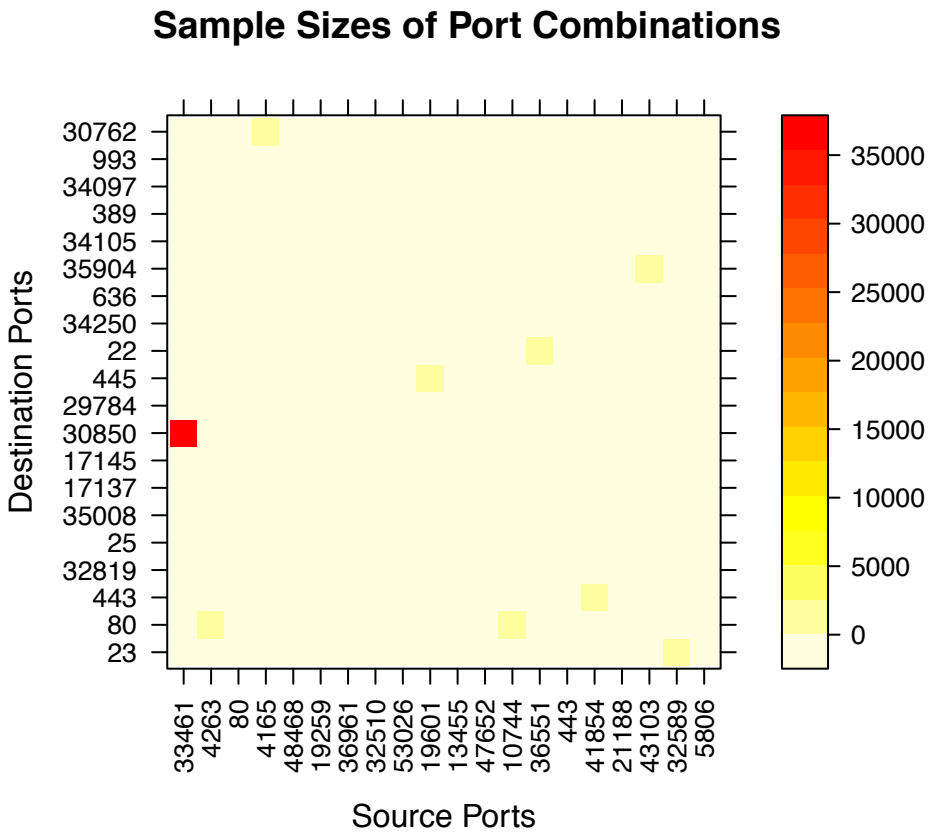
The bar plots above represent the row and column means of the continuous features for each slice of the tensor. These row means and column means inform imputation techniques for the missing cells within those respective rows and columns. There exist clear outliers in the means for certain ports. These outliers may cause problems with imputing missing values in that row or column combination because the outliers will tend to have a larger effect on the imputed value than other means that are more close to the median in the imputed values row or column. This further suggests imputation techniques that take into account variances among the row and column means and the number of samples observed for each port combination will result in more accurate imputations.

2.2.4 Port Connections



The above ports network graph displays the pairings between the top source ports (red cirlces) and the the top destination ports (blue squares). The size of each node reflects the number of paired observations that were observed using that particular port. It is important to note that not every source port is paired with a destination port and vice versa (not every node is connected to every other node). These missing combinations reflect missing values in the Y tensor, and consequently they correspond to the combinations that require values to be imputed.

2.2.5 Matrix Slice Properties



Finally, to better visualize the sample sizes first displayed in the network graph (section 2.2.4), the above heat plot represents the sample sizes for each source-destination port combination. It is again clear there are certain combinations that have many observations (range of 35000), while most of the observations are 0, indicating no observations were observed and the corresponding cell in Y is missing, or near 0, indicating few observations were observed. The large variation in sample sizes suggests an imputation technique that accounts for sample size of a missing cell's related row and column cells is necessary.

Chapter 3

Alternating Least Squares Applied to Two Dimensional Matrices

3.1 Motivation

The preliminary tensor imputation technique slices the tensor into four separate matrices, $Y^{(1)}, Y^{(2)}, Y^{(3)}, Y^{(4)} \in \mathbb{R}_{m \times n}$. Each matrix has dimensions defined by the number of source ports by the number of destination ports. Each cell, $Y_{ij}^{(k)}$ contains the mean of the observations observed for continuous feature k and source port i destination port j. Each matrix will have its missing cells imputed with information from the rest of that matrix, and separately from information about the other matrices. The four completed matrices will then be recombined to create the tensor Y. Similar techniques for matrix completion were employed in the Netflix Challenge where top competitor predicted ratings for movies by users that had not watched the movie based on the other ratings in the matrix of users and movies.

3.2 Matrix Completion Algorithm

Each $Y^{(k)}$ has missingness because not every source port interacts with every destination port. $F \in \mathbb{R}_{m \times n}$ is a sparse matrix that represents the frequencies of combinations, i.e $F[32242, 12312]$ represents the number of observations for the 32242 12312 port interaction. $M \in \mathbb{R}_{m \times n}$ represents a boolean matrix of whether the corresponding Y values are missing. $Y^{(k)}[M]$ represents all of the missing values of $Y^{(k)}$. Moreover because each non-missing observation contains all four continuous features, $Y[M]$ represents all the missing values of Y.

The objective is

$$min \sum_{i,j:F_{i,j}>0} (Y_{i,j}^{(k)} - u_i^{(k)} D^{(k)} v_j^{(k)T})^2$$

where $U^{(k)}D^{(k)}V^{(k)T}$ represents the singular value decomposition of $Y^{(k)}$. There are multiple steps to the matrix completion process:

3.2.1 ANOVA Initial Imputation

Impute the initial values for the missing $y_{i,j}^{(k)}$ observations $1 \leq i \leq m, 1 \leq j \leq n$: In general an additive model is applicable:

$$y_{i,j}^{(k)} = \mu^{(k)} + a_i^{(k)} + b_j^{(k)} + \epsilon_{i,j}^{(k)}$$

where $\epsilon^{(k)} \in N(0, \sigma^2)$, $\mu^{(k)}$ is the overall mean of $Y^{(k)}$, $a_i^{(k)}$ is the row mean, and $b_j^{(k)}$ is column mean. An analysis of variance (ANOVA) imputation is used to fill in the initial values, $y_{i,j}^{(k)}$. Ignoring the missing values for now, let $y_{..}$ denote the empirical overall mean, $y_{i.}$ denote the empirical row mean, and $y_{.j}$ denote the column mean.

$$y_{i,j} = y_{..} + (y_{i.} - y..) + (y_{.j} - y_{..}) = y_{i.} + y_{.j} - y..$$

3.2.2 Repeated Imputation

The repeated imputation procedure solves $Y^{(s)}[M] = R_k(Y^{(s-1)})[M]$ where R_k is the best rank-k approximation for the s-th step. For each step (s) use singular value decomposition to decompose

$$Y^{(s)} = U^{(s)}DV^{T(s)}$$

where D is a diagonal matrix of the singular values, U is the left singular vectors of Y and V is the right singular vectors of Y.

The Eckart-Young-Mirsky (EYM) Theorem provides the best rank-k approximation for the missing values in $Y^{(s+1)}$. Recall $Y[M]$ represents all of the missing values of Y. Applying the EYM theorem:

$$Y^{(s+1)}[M] = (U[, 1:k]^{(s)}D[, 1:k]V[, 1:k]^{T(s)})[M]$$

Where $U[, 1:k]$ represents the first k columns of U and the same for D and V.

3.2.3 Convergence Criterion

The EYM rank approximation imputation steps are repeated until the relative difference between $Y^{(s+1)}$ and $Y^{(s)}$ falls below a set threshold, T. The relative difference threshold is expressed:

$$\frac{\|Y^{(s+1)} - Y^{(s)}\|_2}{\|Y^{(s)}\|_2} < T$$

where $\|Y\|_2$ is the Frobenius norm. The denominator of the expression ensures the convergence criterion is invariate to a scale change in the matrix itself.

3.2.4 Leave One Out Cross Validation

To assess the quality of the imputation, Leave-One-Out Cross Validation (LOOCV) is used to generate a prediction error. LOOCV cycles through the observed values, setting each to NA (missing), and then performing the described imputation process. The prediction error is then calculated as some function of the difference between the imputed value and the true value. In this case, the algorithm records absolute error $\sum \mid \hat{y}_{i,j} - y_{i,j} \mid$ and root mean square error $\sqrt{\frac{\sum(\hat{y}_{i,j}-y_{i,j})^2}{n}}$ where n is the number of observations not missing.

3.3 Validation Against Simulated Data

Before applying the algorithm on the real data it is useful to validate the algorithmic approach against simulated data where the true rank is known.

3.3.1 Simulating a Low Rank Matrix

Taking the Kronecker product of two lower dimension matrices yields a higher dimension matrix with low rank. Explicitly, given matrix $A \in \mathbb{R}_{m\times r}$ and $B \in \mathbb{R}_{r\times n}$, $A \otimes B = C$ where $C \in \mathbb{R}_{m\times n}$ with rank r. Thus, when $r < m, r < n$ the matrix C has an optimal low rank that minimizes the root mean square error from the leave one out cross validation procedure. To add noise to the simulated matrix, C, simply add an error matrix, $E \in \mathbb{R}$ where $e_{i,j} \sim N(0, 1)$.

Moreover, this procedure provides a computationally efficient way to simulate many random low rank matrices to use as inputs for the validation procedure. In the case of simulated matrices, there are no missing entries, so the leave one out cross validation procedure sequentially removes each cell in the matrix, imputes its value using the rank being investigated, and considers the individual cell error as the difference between the true value and the imputed value. The overall root mean square error for the technique is then calculated with the aggregate each of these individual cell errors.

3.3.2 Approximating Optimal Rank

WHAT PLOTS OR VISUALIZATIONS DO I USE FOR RESULTS? HISTOGRAM OF PERCENTAGES FOR RANK APPROXIMATIONS? 1-4 100%, higher ranks arent that relevant; effects of noise

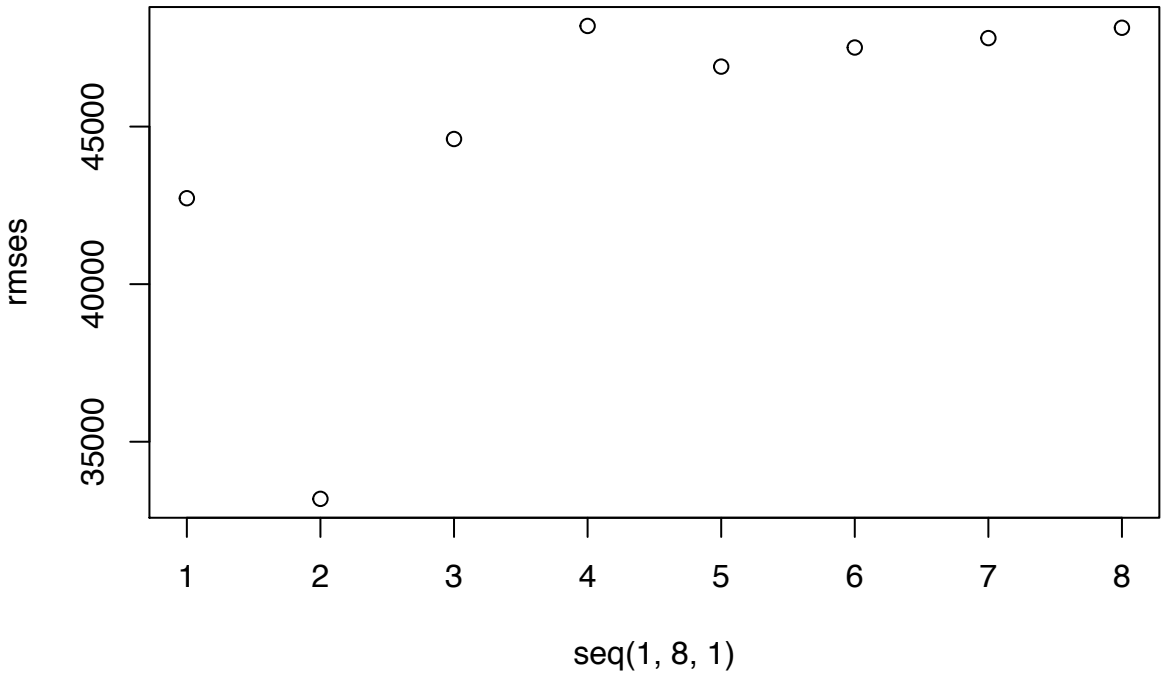
USE THIS TECHNIQUE TO SHOW DIFFERENT EFFECTS OF NOISE?

The above plots represent the accuracy of the matrix completion technique for matrices with true rank $k = 1, 2, ...8$. The accuracy is measured by simulating 10 random

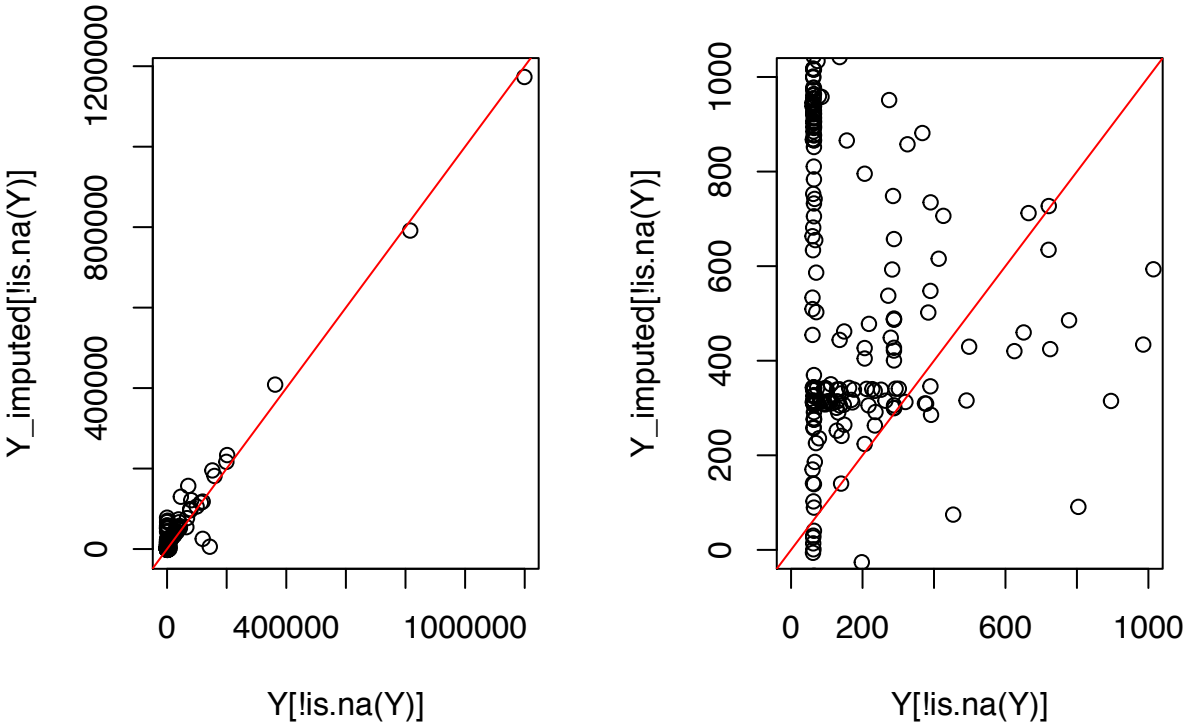
matrices with low rank of k for each value of k (80 simulated matrices total), and then running the leave one out cross validation procedure described above on the matrix C to generate a root mean square error for each possible rank. The accuracy is the calculated using the number of times the rank with the lowest error matches the true simulated rank divided by 10. Note as the true rank becomes larger, the technique performs far worse at determining the optimal low rank, THIS IS BECAUSE_____

When a noise matrix E is added to each simulated matrix C

3.4 Results on Real Data



The above plot displays the root mean square errors from leave one out cross validation across different rank inputs into the algorithm. It's clear that rank 2 provides the best low-rank solution for the alternating least squares imputation algorithm. Thus rank 2 is used in the algorithm to impute the missing values in Y.



The two plots above display the true values of the Y matrix (i.e. the non-missing values) versus their corresponding fitted values using the alternating least squares algorithm with an input of rank 2. The first plot displays all values and shows a somewhat positive linear trend (an ideal fit of the true values would be a scatter of points following the 45 degree angled line represented in red). However, several outliers with large true values skew this dataset and cause the plot to appear linear. Closer examination of the true and fitted values smaller than 1000 (the plot on the right) reveals the relationship is far from the linear pattern.

DISCUSS THE NEED TO HAVE DATA WHERE THE ENTIRE ROW OR COLUMN CANT BE MISSING

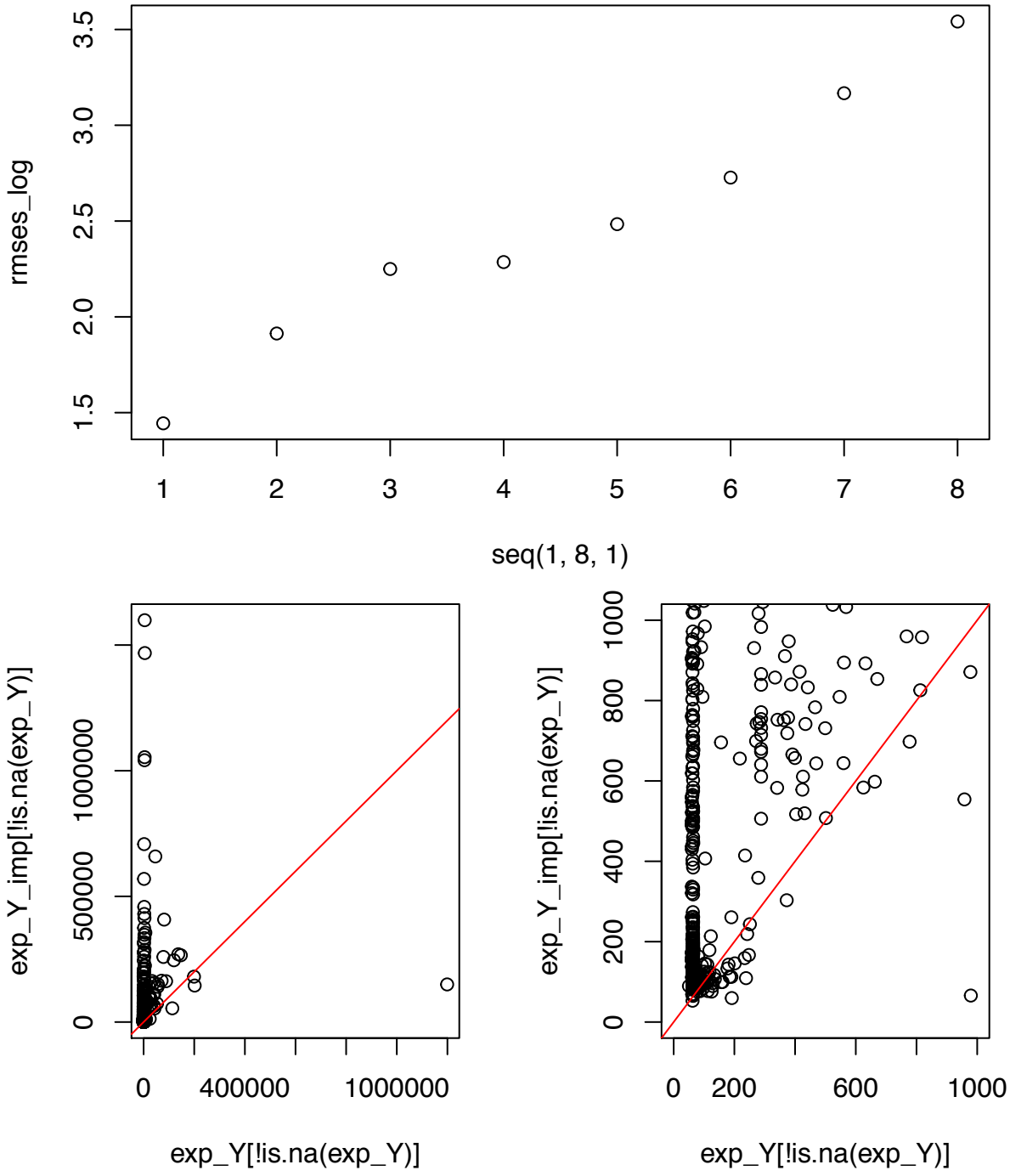
3.4.1 Scale Transformations

The poorly fitted results motivates a consideration of the scale of the data. The present algorithm uses the sample averages of the overall matrix as well as the row and column means when imputing each missing cell value. This reliance upon sample means leads to susceptiblity to outliers. Moreover exploratory data analysis reveals the dataset contains outliers, particularly in the SrcBytes and DstBytes measurements. Thus, a transformation of features in the model may be appropriate for improving the fit of the algorithm.

OUTLIERS DRIVE SUM OF SQUARES

When a natural log transformation is applied to the raw dataset before any imputation steps are taken, the alternating least squares imputation algorithm yields the following

root mean square errors varied by rank.



Chapter 4

A Bayesian Approach to Two Dimensional Matrix Completion

4.1 Motivation

The previous section's results reflected the need for an imputation strategy that accounted for the variability in the number of observations observed for each port combination when imputing that particular combination's cell. The previous technique fails to take into account the differing sample size and variance in each cell (source port i, destination port j), so the algorithm treated each y_{ij}^k, which represents the mean of all observations in cell (i, j) for continuous feature k, as a single value with no consideration of differing sample sizes and variances between cells. The following section constructs a statistical model that takes frequency of observations and variances for each cell into account and repeatedly samples from that statistical model to complete the matrix.

4.2 Statistical Model for Port Relationships

The following statistical model is defined for imputing port relationships:

$$y_{i,j} = u_i^T v_j + \frac{\sigma_i \tau_j}{\sqrt{n_{ij}}} \epsilon_{ij}$$

where u_i represents _____, v_j represents _____, σ_i represents the MEAN??? of the standard deviations of each row in the matrix, τ_j represents the MEAN??? of the standard deviations of each column in the matrix, and $\epsilon_{ij} \sim N(0, 1)$. Fixing the j values in the analysis (i.e. v_j and τ_j are known) enables the model to be rewritten in the form of a weighted least squares model for imputing u_i and σ_i. Similarly, when i is fixed, the model can be rewritten to simulate v_j and τ_j. In particular, fixing the

jth row of the matrix yields:

$$y_i = \beta^T x_i + \sigma w_i \epsilon_{ij}$$

where $w_i = (\frac{\tau_j}{\sqrt{(n_{ij}}})^2$, and $\beta^T = u_i$. Note, this technique again slices the tensor into the four separate matrices, $Y^{(1)}, Y^{(2)}, Y^{(3)}, Y^{(4)} \in \mathbb{R}_{m \times n}$, and the model can be applied to each matrix $Y^{(k)}$ separately.

Vectorizing the above model yields

$$y_i = X\beta + \sigma W^{1/2} \epsilon$$

where $W \in \mathbb{R}_{m \times m}$ is the diagonal matrix of weights, such that

$$W = \begin{bmatrix} w_1 & & \\ & \ddots & \\ & & w_m \end{bmatrix} = \begin{bmatrix} \frac{\tau_1^2}{n_{11}} & & \\ & \ddots & \\ & & \frac{\tau_n^2}{n_{mn}} \end{bmatrix}$$

X is set as the matrix of u_i's for all i if j is fixed or the matrix of v_j's for all j if i is fixed.

Note X and W do not depend on i, so they are used in the imputation for u_i and σ_i.

This is a specific form of the Generalized Least Squares Model (GLS), which gives a weighted least squares estimate of β, and it is appropriate when the error terms are not independent and identically distributed. Bayesian analysis of this problem provides similar parameter estimates to GLS, and both ordinary least squares and GLS provide unbiased parameter estimates of β with the latter giving estimates with a lower variance because the non-Bayes estimator serves as a limit of the Bayes estimator.

The full conditional distributions of the random variables β and σ^2 for this case of GLS are described below:

$$\{\beta \mid X, \vec{y}, \sigma^2\} \sim MVN(\beta_n, \Sigma_n)$$

$$\{\sigma^2 \mid X, \vec{y}, \beta\} \sim IG(\frac{\nu_0 + m}{2}, \frac{v_0 \sigma_0^2 + SSR_W}{2})$$

where MVN represents the Multivariate Normal Distribution, IG represents the Inverse Gamma distribution, W is described above, and

$$\begin{cases} \Sigma_n = (X^T W^{-1} X/\sigma^2 + W_0^{-1})^{-1} \\ \beta_n = \Sigma_n (X^T W^{-1} y/\sigma^2 + W_0^{-1} \beta_0) \end{cases}$$

$$SSR_W = (y - X\beta)^T W (y - X\beta)$$

The remaining variables in the closed form full conditionals come from the random variables prior distributions, which are defined as follows:

$$\beta \sim MVN(\beta_0, W_0)$$

$$\sigma^2 \sim IG(\frac{\nu_0}{2}, \frac{\upsilon_0}{2}\sigma_0^2)$$

The initial values for the prior distributions are set as: $\beta_0 = 0$, $W_0 = \gamma^2 I$ where γ^2 is a large number and I is the $m \times n$ identity matrix, $\nu_0 = 2$, $\sigma_0^2 = 1$. This results in a diffuse prior for β that spreads out the density, and a noninformative prior for σ_0^2.

SHOW THAT GENERALIZED LINEAR MODEL IS (Xˆ T Wˆ -1 X)ˆ -1 X$^{\text{TW}}$-1y

The derivation of the general case can be found in APPENDIX????

4.2.1 Generalized Gibbs Sampler

Following the formulation of the model and the definition of the priors, a Gibbs Sampler is created to repeatedly generate samples from the full conditional of each parameter in the statistical model, which iteratively creates an approximate value for each cell.

The Gibbs Sampler algorithm progresses as follows:

Let the set of parameter samples at step s in the algorithm be defined as $\phi^{(s)} = \{\beta^{(s)}, \sigma^{2(k)}\}$.

1. Sample $\beta^{(s+1)} \sim P(\beta \mid X, \vec{y}, \sigma^{2(k)})$
2. Sample $\sigma^2 \sim P(\sigma^2 \mid X, \vec{y}, \beta^{(s+1)})$

Set $\phi^{(s+1)} = \{\beta^{(s+1)}, \sigma^{2(k+1)}\}$

This Gibbs Sampler serves as a general technique that can be used to simulate both the values of u_i and σ_i or v_j and τ_j depending on the inputs it is given because the formulation of both models are identical; they only differ by the calculations of the inputs, X and W. In the context of the problem, it can first be called to simulate all of the rows of the matrix $Y^{(k)}$, then called to simulate all of the columns of the matrix, and the process repeats from there.

4.3 Procedure for Imputing Missing Values

Using the generalized Gibbs Sampler it is possible to define a procedure for iteratively simulating missing values for the entire matrix $Y^{(k)}$.

1. Initialize σ_i and τ_j as the overall standard deviation of the $Y^{(k)}$ matrix.

2. Simulate u_i and σ_i using the generalized Gibbs Sampler. Set random values for the starting value of X, the algorithm will naturally converge to the true values of v_j.
3. Simulate v_j and τ_j using the generalized Gibbs Sampler. Set random values for the starting value of X, the algorithm will naturally converge to the true values of u_i.
4. Fill in the missing values y_{ij} in $Y^{(k)}$ by sampling from the normal distribution

$$y_{ij} \sim N(u_i^T v_j, \frac{\sigma_i^2 \tau_j^2}{\sqrt{(n_i j)}})$$

Chapter 5

Tensor Completion

5.1 Imputation Strategy

The imputation strategy focuses on finding a low rank approximation for Y when decomposing the tensor.

5.1.1 CP Decomposition

The CP decomposition expresses the tensor as:

$$Y = \sum_{r=1}^{R} u_r \cdot v_r \cdot w_r$$

where r represents the rank approximation, $\cdot$ denotes the outer product of tensors, and $u \in \mathbb{R}_{\gg \times \diagdown}$, $v \in \mathbb{R}_{\ltimes \times \diagdown}$, and $w \in \mathbb{R}_{\not\geq \times \diagdown}$. Each individual cell is expressed:

$$y_{ijk} = \sum_{r=1}^{R} u_{ri} \cdot v_{ri} \cdot w_{ri}$$

Applying this decomposition yields the objective

$$min_{Y'} \|Y - Y'\|, Y' = \sum_{r=1}^{R} \lambda_r (u_r \cdot v_r \cdot w_r)$$

, where $lambda_r$ is the regularization penalty.

5.1.2 Variable Sample Sizes

A traditional approach to tensor completion involves using alternating least squares regression to impute the missing values after populating them with some initial values.

The previous section applies this approach to a 2-dimensional $m \times n$ tensor that represents a cross-section slice of Y that only includes one of the four continuous features. *include als section here, related work:* `https://arxiv.org/abs/1410.2596` *(hastie fast als), application netflix challenge*

While this approach yields a completed tensor, it does not account for the fact that the means in each cell are calculated from a variable number of observations. Furthermore it is not necessarily true that $n_{ijk} = n_{i'j'k'}$ or $\sigma^2_{ijk} = \sigma^2_{i'j'k'}$ for $i \neq i', j \neq j', k \neq k'$.

We propose the following model:

$$y_{ijk} \sim N(\mu_{ijk}, \frac{\sigma^2_{ijk}}{n_{ijk}})$$

where μ_{ijk} is the sample mean, n_{ijk} is the sample size, and σ^2_{ijk} is the sample variance of observations for source port i, destination port j, and continuous feature k.

Substituting these values into the Gaussian probability density function yields the likelihood:

$$\frac{n_{ijk}}{\sigma^2_{ijk}} \sum (\bar{y}_{ijk} - \mu_{ijk})^2$$

Applying the CP/PARAFAC decomposition u_{ijk} is re-expressed:

$$u_{ijk} = \sum_{r=1}^{R} a_{ir} b_{jr} c_{kr}$$

Vectorizing the inputs in the likelihood yields:

$$\sum_j \sum_k [\bar{y}_{ijk} - a_i^T (b_i \cdot c_k)] \frac{n_{ijk}}{\sigma^2_{ijk}} (1)$$

where $a_i \in \mathbb{R}_{\gg \times \backslash}$, $b_j \in \mathbb{R}_{\ltimes \times \backslash}$, and $c_k \in \mathbb{R}_{\not\geq \times \backslash}$. Summing across j and k in this case solves for the ith row slice of the tensor. How to notate vectorization of y?

Recall the Residual Sum of Squares (RSS) of the likelihood for an Ordinary Least Squares (OLS) regression is expressed:

$$\sum_l (y_l - B^T x_l)^2$$

Adding a weight, w_l to the summation yields a Weighted Least Squares problem (WLS)

$$\sum_l^n w_l (y_l - \beta^T x_l)^2 (2)$$

that is analagous to the vectorized likelihood equation (1) with $w_l = \frac{n_{ijk}}{\sigma^2_{ijk}}$, $\beta = a_i$, $x = (b_j \cdot c_k)$.

With this formulation its now possible to solve for the optimal values for each slice a_i of the tensor.

Recall that in a traditional vectorized OLS, $y = X\beta + \sigma\epsilon$, where $y \in \mathbb{R}_{\aleph \times \not\exists}$, $X \in \mathbb{R}_{\aleph \times \imath}$, $\beta \in \mathbb{R}_{\imath \times \not\exists}$, and
$sigma\epsilon \in \mathbb{R}_{\aleph \times \not\exists}$. Solving the maximum likelihood estimator of β, gives $\hat{\beta} = (X^T X)^{-1} X^T y$.

Applying this formulation to the weighted least squares gives $y = X\beta + W^{-\frac{1}{2}}\epsilon$. Solving for the weighted least squares estimator gives $\hat{\beta} = (X^T W X)^{-1} X^T W y$.

Repeating this estimation technique across each slice of the tensor a_i, b_j, c_k results in a completed model for y_{ijk}.

5.2 Gibbs Sampling

Following the formulation of the model, a Gibbs Sampling algorithm is used to repeatedly generate samples from the full conditional of each parameter in the model statistical model, which iteratively creates an approximate value for each cell.

Conclusion

If we don't want Conclusion to have a chapter number next to it, we can add the `{-}` attribute.

More info

And here's some other random info: the first paragraph after a chapter title or section head *shouldn't be* indented, because indents are to tell the reader that you're starting a new paragraph. Since that's obvious after a chapter or section title, proper typesetting doesn't add an indent there.

References

Angel, E. (2000). *Interactive computer graphics : A top-down approach with opengl*. Boston, MA: Addison Wesley Longman.

Angel, E. (2001a). *Batch-file computer graphics : A bottom-up approach with quicktime*. Boston, MA: Wesley Addison Longman.

Angel, E. (2001b). *Test second book by angel*. Boston, MA: Wesley Addison Longman.