Multi-omic Network Regression: Methodology, Tool and Case Study

Vandan Parmar, Pietro Lió

Abstract The analysis of biological networks is characterized by the definition of precise linear constraints used to cumulatively reduce the solution space of the computed states of a multi-omic (for instance metabolic, transcriptomic and proteomic) model. In this paper, we attempt, for the first time, to combine metabolic modelling and networked Cox regression, using the metabolic model of the bacterium *Helicobacter Pylori*. This enables a platform both for quantitative analysis of networked regression, but also testing the findings from network regression (a list of significant vectors and their networked relationships) on *in vivo* transcriptomic data. Data generated from the model, using flux balance analysis to construct a Pareto front, specifically, a trade-off of Oxygen exchange and growth rate and a trade-off of Carbon Dioxide exchange and growth rate, is analysed and then the model is used to quantify the success of the analysis. It was found that using the analysis, reconstruction of the initial data was considerably more successful than a pure noise alternative. Our methodological approach is quite general and it could be of interest for the wider community of complex networks researchers; it is implemented in a software tool, MoNeRe, which is freely available through the Github platform.

1 Introduction

Large scale networks have increasingly become ubiquitous with the development of the Internet of Things [1], the smart grid [2, 3] and smart motorways [4]. These networks have also been developed for biological purposes, both to identify significant genes for the growth of cancer (networked Cox regression [5]) and model whole cell behaviour (metabolic modelling [6]).

Both networked Cox regression and multi-omic metabolic modelling are new techniques which aim to develop the more established Cox regression and metabolic modelling. Cox regression attempts to find significant genes within a dataset of gene expression data. Previously this would have only been found by taking samples from patients or bacteria. However, with multi-omic metabolic modelling, gene expressions can be incorporated into the modelling framework, meaning gene expression data can be generated computationally using these models. Where normally it is difficult to verify the effectiveness of Cox regression, when the gene expression data is created using a computational model, the same computational model can be used to test the predictions of the Cox regression.

This paper focuses on the metabolic network of *Helicobacter Pylori*, a bacterium usually found in the stomach. Originally found in 1982 to be causing stomach ulcers [7] and linked to stomach cancer [8], *Helicobacter Pylori* causes no symptoms in approximately 80% of infected individuals and is thought to be important for the stomach ecology [9].

Vandan Parmar
University of Cambridge, e-mail: vandan.parmar@googlemail.com

Pietro Lió
University of Cambridge, e-mail: pl219@cam.ac.uk

Datasets were created using the *H. Pylori* metabolic model, the regression was analysed and tuned, and the success of the overall regression was then tested by attempting to reconstruct the original data. A visual representation of the developed work flow is shown in figure 1. The following section contains a summary of related research, section 3 details the theory and computational methods required, results and discussion is provided in section 4 and concluding remarks are given in section 5. The code from this paper can be found at: `https://github.com/vandanparmar/MoNeRe`.

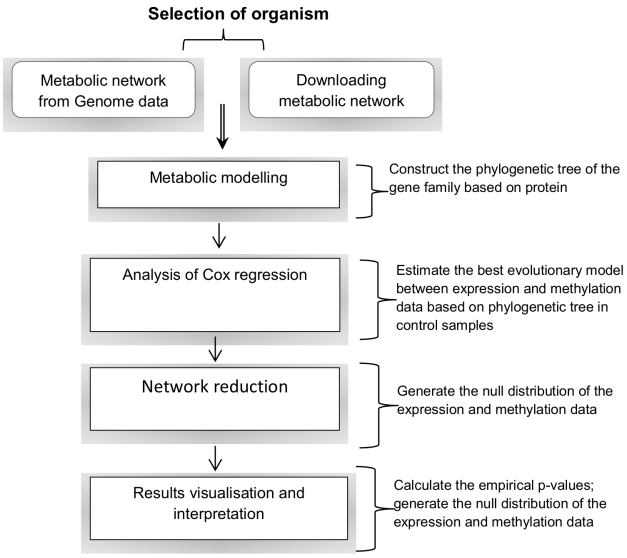

Fig. 1: A visual representation of the work flow developed in this project.

2 Background

2.1 Metabolic Modelling

The emergence of new sequencing methods allowing for complete organism genomes to be sequenced and the explosion of computing power over the past 30 years, has lead to the emergence of Systems Biology as a field of study [10]. This has allowed increasingly good models of a cell to be developed. Specifically, models of a cell's metabolism (the set of chemical reactions taking place in the cell) can be made. The first was created by Tomita et al. [11], however more recent metabolic models are designed to include data from other sources. These include the proteome (set of proteins expressed by a genome) and the transcriptome (set of messenger RNA molecules expressed by the genome) and are collectively known as 'omic' data. A multi-omic model of this type can be predictive [12], often aiming to find unexpected behaviour which can then be tested *in vitro*.

Once a model is constructed, analysis is primarily carried out using Flux Balance Analysis (FBA). There are several existing toolboxes which provide frameworks for carrying out FBA on genome scale metabolic models, including COBRA [13], METRADE [12] and DFBALab [14], in this study, COBRA will be used for FBA calculations. FBA is a particularly powerful method, as it reduces the multidimensional search for an optimal metabotype (set of fluxes through all reactions in the system) to a linear program [15], under the assumptions of a steady state. FBA has been used to model bacteria such as *Escheria Coli* [16] and more recently breast cancer cells [17].

2.2 Cox Regression

Cox regression [5] was created as an extension of Kaplan-Meier survival analysis [18]. Kaplan-Meier survival analysis aims to find the probability of survival for a species whilst incorporating data from living individuals. The Cox model furthers this by including extra data (other than survival time) about each individual. This data could be anything, but is most often gene expression data. More recently, this has been extended by adding network constraints [19] both to enhance the regression and discover networks relating to the regression. This has been used successfully identify significant biomarkers in predicting the outcome of Ovarian cancer treatment [20].

3 Theory and Computational Methods

3.1 Network Regression

Network regression [21] is the primary method of Pareto data analysis used in this study. Compared to linear regression, networked constraints are imposed on regression coefficients, enforcing similarity between the regression coefficients of linked genes.

Cox regression [5] is used as a basis for the networked regression used. The Cox hazard model assumes a hazard function, $h(t|\mathbf{X}_i)$, the risk of death at time t,

$$h(t|\mathbf{X}_i) = h_0(t)\exp(\mathbf{X}_i \cdot \boldsymbol{\beta}) \tag{1}$$

where $\mathbf{X}_i$ is the ith gene expression profile, $h_0(t)$ is the baseline hazard and $\boldsymbol{\beta}$ is the vector of regression parameters.

Within the Cox model, the regression parameters are estimated by maximizing the Cox log-partial likelihood,

$$p_l(\boldsymbol{\beta}) = \sum_{i=1}^{n} \delta_i \left\{ \mathbf{X}_i \cdot \boldsymbol{\beta} - \log \left[\sum_{j \in \mathscr{R}(t_i)} \exp(\mathbf{X}_j \cdot \boldsymbol{\beta}) \right] \right\} \quad (2)$$

where $\delta_i = 0$ if the ith individual is alive and $\delta_i = 1$ if the ith individual is dead and $\mathscr{R}(t_i)$ is the set of individuals alive at time t_i.

The network constraints can be added in a variety of ways [19], in this study a networked penalty, $p_n(\boldsymbol{\beta})$, was subtracted from the log-partial likelihood,

$$p_n(\boldsymbol{\beta}) = -\lambda\alpha|\boldsymbol{\beta}|^2 - \lambda(1-\alpha)\boldsymbol{\beta}^T\mathbf{S}\boldsymbol{\beta} \quad (3)$$

where λ and α are tuning parameters, $\mathbf{I}$ is the identity matrix and $\mathbf{S}$ is a weight matrix related to the network. The second term is the network penalty itself and the first term is a sparsity constraint, forcing the regression to identify only a few significant genes. α controls the ratio between the sparsity and network constraints and λ controls the ratio between the simple Cox regression and the networked constraints.

$\mathbf{S}$ is computed by first computing $\mathbf{W}$, a gene coexpression network,

$$w_{ij} = \mathrm{PMCC}(g_i, g_j) \quad (4)$$

$$(\mathbf{W})_{ij} = \begin{cases} w_{ij}^2 & w_{ij}^2 > \zeta \\ 0 & w_{ij}^2 < \zeta \end{cases} \quad (5)$$

where PMCC indicates the product moment correlation coefficient, so the values of $\mathbf{W}$ are between ζ and 1. The value of ζ can be tuned to change the connectivity of the network. The adjacency matrix of the network is thus the ceiling of $\mathbf{W}$. From this $\mathbf{S}$ can be calculated,

$$(\mathbf{S})_{ij} = \begin{cases} 1 & i = j \text{ and } d_i \neq 0 \\ \dfrac{-(\mathbf{W})_{ij}}{\sqrt{d_i d_j}} & (i,j) \in \mathscr{E} \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

where d_i is the degree of the ith node and $\mathscr{E}$ is the set of edges within the network. $\mathbf{S}$ is a weighted version of the normalized Laplacian of a graph [22] and this matrix is always positive semidefinite (i.e. $\boldsymbol{\beta}^T\mathbf{S}\boldsymbol{\beta} \geq 0$).

As $\mathbf{S}$ is always positive semidefinite, all terms in the penalty function $p_l(\boldsymbol{\beta}) - p_n(\boldsymbol{\beta})$ are convex (surprisingly $\log\sum_i \exp(\mathbf{X}_i \cdot \boldsymbol{\beta})$ is convex [23]), meaning that convex optimization [24] can be used to perform the regression.

In the case of a Pareto front, we perform regression on each side of the phase transition individually, considering individuals in one phase to be alive and those in the other dead. The optimization can then be simply formed as,

$$\begin{aligned} &\max \{p_l(\boldsymbol{\beta}) - p_n(\boldsymbol{\beta}; \lambda, \alpha)\} \\ &\text{s. t.} \qquad |\boldsymbol{\beta}| \leq 1 \end{aligned} \quad (7)$$

where x_p is the value of an objective function at the phase transition and x_j is the corresponding objective value of the individual.

3.2 Network Reduction

When reconstructing Pareto fronts after completing regression, to further impose network constraints, a subsection of the relevant network containing significant nodes can be found. This was done by creating a subset of the network where each node is at a maximum of one edge from a node in the subset as shown in figure 2. The procedure for this is as follows,

1. Create a new copy of the graph (to edit)
2. Create a list of nodes, sorted by degree descending.
3. Remove the first node in the sorted list, add it to the reduced network, add its neighbours to a list of accounted for nodes, remove the node from the graph.
4. Update the sorted list of nodes.
5. Repeat steps 2 and 3 until all nodes are accounted for.

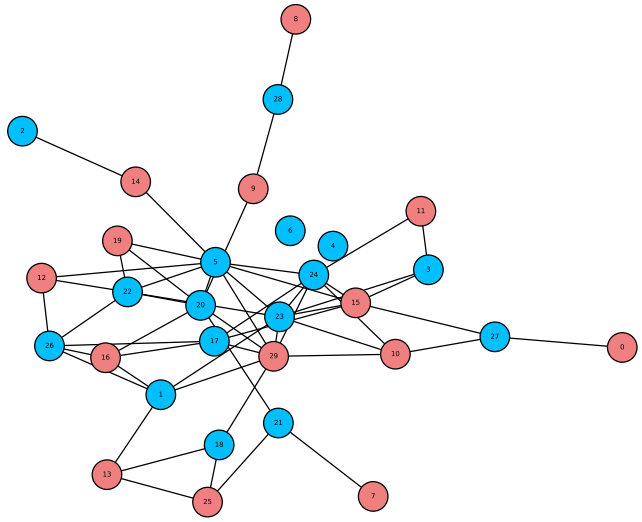

Fig. 2: A network showing nodes included in the reduced network in blue and others in red. Red nodes are connected to at least one blue node.

3.3 Implementation Outline

The code for the project was exclusively written in Python3, however, multiple libraries were used in conjunction. The external libraries used were: Numpy, Networkx, CVXpy, COBRA, DEAP, TQDM. Networkx is used for construction and calculation on networks, CVXpy for the network regression, COBRA for handling the metabolic model, DEAP for the genetic algorithm and TQDM for providing progress bars for code with longer run times. The *H. Pylori* model [25] used is taken from BiGG Models [26].

The code for this paper is available at: `github.com/vandanparmar/MoNeRe`.

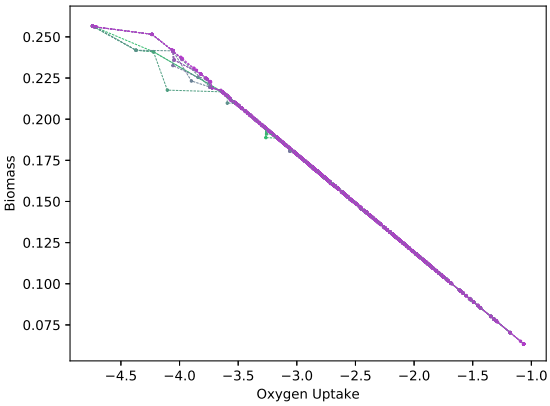

Fig. 3: Graph showing the progression of the genetic algorithm, forming the Pareto front between Oxygen uptake and Biomass (cell growth rate). Points from earlier generations are in green, points from later generations are in purple. The general progression is to the upper right quartile, the non dominated area.

4 Results and Discussion

4.1 Data Generation

The network of focus in this study is the metabolic network of *H. Pylori*. To see the way the genetic algorithm works, the Pareto front for the trade-off between Biomass (the amount of cell growth) and Oxygen uptake was found, shown in figure 3. After exploring several trade-offs, this trade-off was chosen as it gives a particularly nice phase transition. The Oxygen, biomass trade-off is also closely linked to energy, thus we thought it likely that many genes would be involved. The values for Oxygen uptake are negative, thus maximising this has the effect of attempting to reduce the amount of Oxygen absorbed by the cell. This graph therefore shows the comparison of growth rate in aerobic compared to anaerobic conditions. We thus might expect that there would be a phase transition between the aerobic and anaerobic regime.

The Pareto front was calculated again, with both more individuals in each generation and more generations, as shown in figure 4. This shows a phase transition, with the Pareto front changing gradient from -0.051 to -0.059 either side of the transition. Transitions like these were found with a range of different objective functions. In order to investigate these, a simplified model was created.

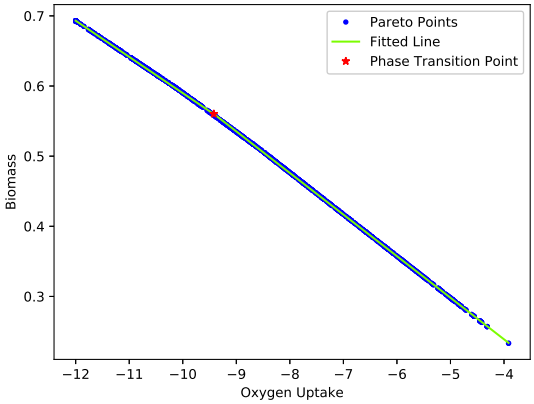

Fig. 4: Graph showing a Pareto front with more points than in figure 3. This Pareto front shows a phase transition, the gradient of the Pareto front changes from -0.051 to -0.059 on either side of the red star indicating the phase transition point.

4.2 Loopless FBA

It is clear that the initial flux bounds used within a model are significant. Changing flux bounds is equivalent to changing the environment (both internal and external) that a cell is operating in. In the literature, FBA is typically used to investigate a specific function or behaviour only, appropriate flux bounds can thus be found from experimentation or to enforce desired conditions [6]. However, in this case, we are looking at arbitrary pairs of objectives, where the large flux bounds used in the *H. Pylori* BiGG model [26] are not appropriate.

To find reasonable upper and lower bounds for each reaction, we use a technique known as Loopless FBA. This adds an additional thermodynamical constraint to FBA, which makes loops or cycles infeasible. Loopless FBA (implemented in the COBRApy toolbox [13]) was used to maximize and minimize the flux through each cell reaction. The maximum and minimum flux through each reaction across all of these runs was then used as the initial

flux bound. In this way we avoid the need for biological data to set specific reaction constraints, though in practice this would still be ideal.

4.2.1 Parameter Comparison

The Cox based network regression, as described in section 3.1, has 3 parameters that can be tuned. λ controls the ratio of the network constraints to the simple Cox regression, α controls the ratio between the network constraint and the sparsity constraint and ζ controls the connectivity of the gene coexpression network. These values can be found using cross-validation.

To analyse the performance of network regression with differing parameters, two test data sets will be used. One from the Oxygen, Biomass trade-off shown at the beginning of section 4, but with the new flux bounds, the other from a Carbon Dioxide, Biomass trade-off. Figure 5 shows the Pareto fronts in both cases. These were chosen as they are very different data sets, the Oxygen trade-off is very dense whereas the Carbon Dioxide trade-off has very few points.

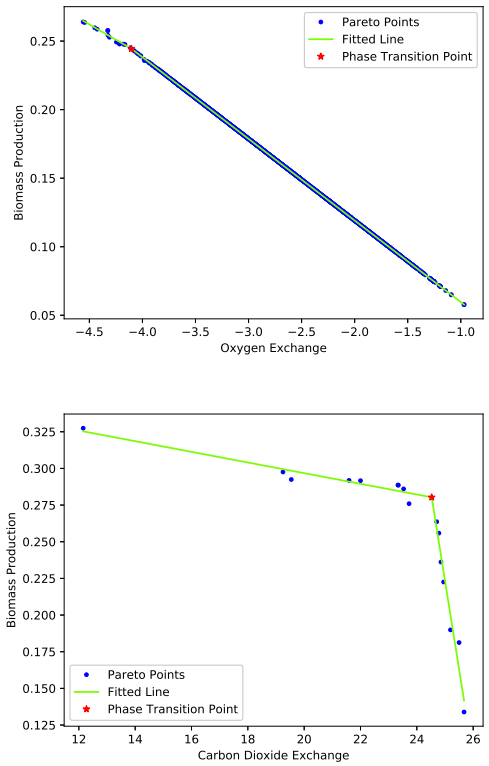

Fig. 5: The Pareto fronts from the Oxygen and Carbon Dioxide trade-offs with Biomass. The Oxygen trade-off is particularly dense, whereas the Carbon Dioxide trade-off is very sparse.

4.3 Pareto Reconstruction

The regression parameters used were ($\alpha = 0.5, \lambda = 1.0, \tau = 99$). From this, the effectiveness of the network regression can be tested. Using exclusively the gene expression data, without knowledge of the metabolic or genetic network, the significant genes can be set to particular values, whilst leaving insignificant genes to take random values. The objectives can then be calculated using the generated gene expression data and compared against gene expression data comprised of pure noise.

The relevant measure of spread for the values of β is the interquartile range. A significant value (outlier larger than the mean) is classed as a value one interquartile range larger than the mean. The network of significant genes is then the network formed from the subsection of the gene coexpression network containing only significant genes. These are shown in figures 8 and 9. The green nodes are those that are more significant and red nodes are less significant. The names used are taken from the metabolic network. Red edges are those from a coexpression network with $\tau = 95$ compared to 99 used in the regression itself which are indicated in blue.

With the significant genes identified, a distribution of noise must be chosen for the insignificant genes. To choose this, we look at the distribution of gene expression values in the original dataset. From figure 6 the distribution of insignificant genes appears to be uniform between 0 and 2.

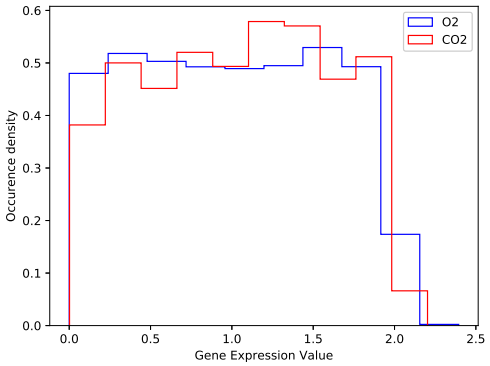

Fig. 6: Histogram of the distribution of gene expression values in the Pareto datasets. This appears to be a uniform distribution between 0 and 2.

Values for the significant genes, are chosen as follows. Key nodes are identified using the network reduction technique described in section 3.2, values from the original data are selected for each of these nodes. Values for other significant nodes are chosen to be values from the orig-

inal dataset corresponding to adjacent key nodes. By doing this the existing distributions of values is kept for each of the significant nodes and the network relations are enforced.

This process gives a new set of gene expression data, the objectives for this data can be calculated and a new Pareto front constructed. These reconstructions are shown in figure 7, there are 800 points constructed from the regression (400 left, 400 right) and 400 points constructed from pure noise. The Oxygen reconstruction is not particularly successful, the random data appears to do as well at constructing the Pareto front as the reconstructed data. However, for points on the left hand side of the Pareto front, the reconstructed data is a more successful. The Carbon Dioxide reconstruction is, however, clearly more successful. The red points constructed from the regression on the left hand portion of the Pareto data are considerably more successful than the green noise points.

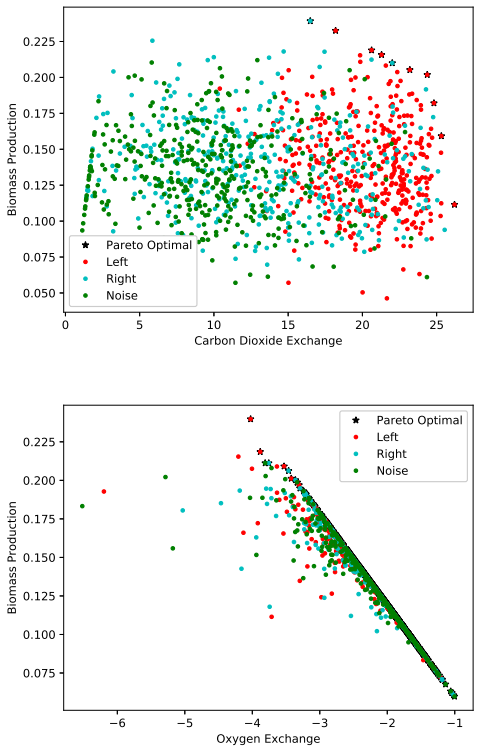

Fig. 7: Reconstructed trade-offs using gene expression created based on the network regression. Points created from pure noise are in green, those from the regression on the left hand side of the Pareto data in red and right are blue. The Carbon Dioxide reconstruction appears to have worked well, whereas points based on Pareto data seem similar to those created from noise for the Oxygen trade-off.

5 Conclusions

Networked Cox regression was used, for the first time, to analyse data from a multi-omic metabolic model of *H. Pylori*. It is clear that the flux bounds used within the metabolic model are significant, this prompted the use of loopless Flux Balance Analysis to choose the flux bounds in this instance.

Two example datasets were then chosen to tune the parameters for the network regression, a Biomass and Oxygen trade-off and a Biomass and Carbon Dioxide trade-off. These were chosen as the Carbon Dioxide trade-off has a small number of points whereas the Oxygen trade-off has a far larger number of points. The regression parameters used were $(\alpha = 0.5, \lambda = 1.0, \tau = 99)$.

With regression variables for each dataset chosen, an attempt was made to reconstruct the original Pareto data. The distribution of gene expression values in the original datasets were used to choose a noise distribution, uniform from 0 to 2. Significant genes were then chosen from the regression variables, those that were one interquartile range above the mean. The reduced network of these genes was then constructed. Significant genes in the reduced network selected values from the original dataset, nodes adjacent to the significant nodes selected values corresponding to those of the adjacent significant nodes. Non-significant genes were left as noise. From this new gene expression datasets were generated and the objectives calculated. The reconstruction for the Oxygen data did not seem to be very different from the noise distribution, except for the values on the left hand side of the Pareto front. In contrast, the reconstruction of the Carbon Dioxide data was successful, with exclusively reconstructed points forming the Pareto front.

In summary we present a computational framework for the integration and analysis of biological network data. Such a challenge can be mapped into three main computational problems: (a) data integration: merging data at different scales (multi-omic data) is necessary to understand the different levels of network evolution and how these levels interact each other; (b) high dimensionality: multi-omic data exists in a high dimensional matrix describing the expression level of each gene or protein; (c) merging statistical and biological knowledge in genomic data analysis: accurate data analysis cannot be performed using only statistical approaches, but rather *a priori* biological knowledge needs to be included in the final computational framework. Indeed, appropriate data investigation should be based both on statistical and biological knowledge in order to merge the information that can be extracted from the data (statistical information) and the biological knowledge (marker related information already known in literature).

The methodology and the software presented in this work could be used for a variety of research useful to the molecular biology community. For example, in bacterial genomes there are high levels of functional redundancies which represent mechanisms employed in cells to achieve robustness. These mechanisms allow, under different environmental conditions, very different sets of reactions to compensate for one another [27]. An interesting application of this method is the possibility of integrating different metabolic networks or different multi omic networks. Another application is the study of the metabolic model of the human (for example from Recon 2) and reconstructions for the dozens of bacterial species forming the gut microbiome.

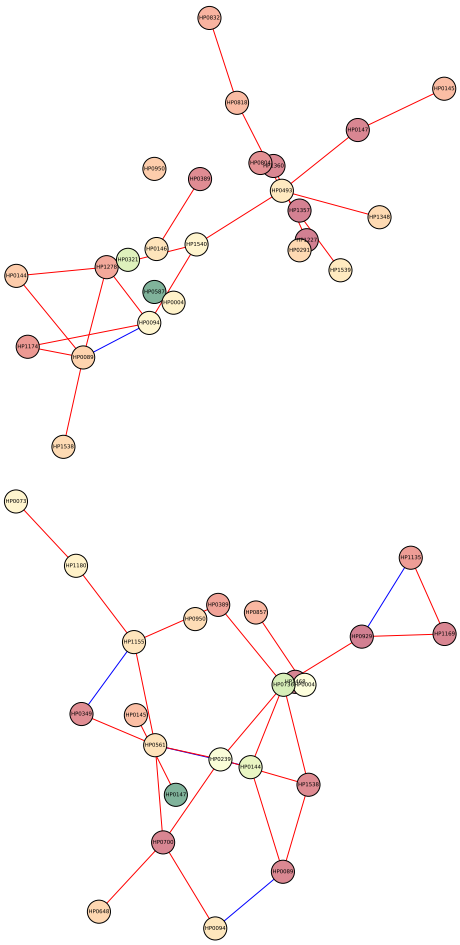

Fig. 8: Significant gene networks for the Carbon Dioxide data. See figure 9 for further explanation.

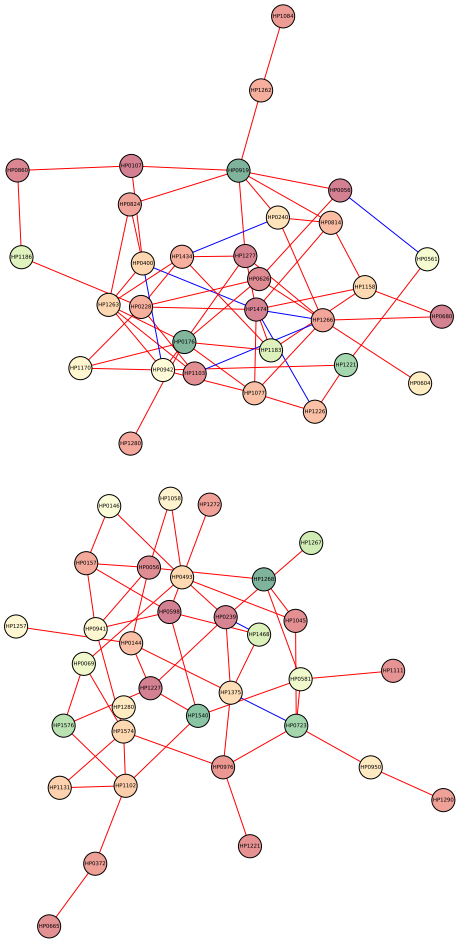

Fig. 9: Significant gene networks for the Oxygen data. Networks for data on the left hand side of the Pareto front are above, and those for the right hand side are below. The colour of nodes indicates how significant the genes are, with green nodes being most significant and red the least significant. The blue edges are those contained within the network constraint, with $\tau = 99$, the red edges are those from the network formed with $\tau = 90$.

References

1. L. Atzori, A. Iera, G. Morabito, Computer Networks **54**(15), 2787 (2010). DOI https://doi.org/10.1016/j.comnet.2010.05.010. URL http://www.sciencedirect.com/science/article/pii/S1389128610001568
2. S.M. Amin, B. Wollenberg, IEEE Power and Energy Magazine **3**(5), 34 (2005). DOI 10.1109/mpae.2005.1507024. URL https://doi.org/10.1109/mpae.2005.1507024
3. H. Farhangi, IEEE Power and Energy Magazine **8**(1), 18 (2010). DOI 10.1109/MPE.2009.934876
4. C.C. Chien, Y. Zhang, P.A. Ioannou, Automatica **33**(7), 1273 (1997). DOI https://doi.org/10.1016/S0005-1098(97)00050-2. URL http://www.sciencedirect.com/science/article/pii/S0005109897000502
5. D.R. Cox, Journal of the Royal Statistical Society. Series B (Methodological) **34**(2), 187 (1972). URL http://links.jstor.org/sici?sici=0035-9246%281972%2934%3A2%3C187%3ARMAL%3E2.0.CO%3B2-6
6. J.D. Orth, I. Thiele, B.Ø. Palsson, Nature Biotechnology **28**(3), 245 (2010). DOI 10.1038/nbt.1614. URL http://www.nature.com/articles/nbt.1614
7. B. Marshall, R. Warren, The Lancet **323**(8390), 1311 (1984). DOI 10.1016/S0140-6736(84)91816-6. URL http://www.sciencedirect.com/science/article/pii/S0140673684918166
8. D. Forman, D.G. Newell, F. Fullerton, J.W. Yarnell, A.R. Stacey, N. Wald, F. Sitas, BMJ (Clinical research ed.) **302**(6788), 1302 (1991). DOI 10.1136/BMJ.302.6788.1302. URL http://www.ncbi.nlm.nih.gov/pubmed/2059685http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC1670011
9. M.J. Blaser, EMBO reports **7**(10), 956 (2006). DOI 10.1038/sj.embor.7400812. URL http://www.ncbi.nlm.nih.gov/pubmed/17016449http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC1618379
10. N. Le Novère, BMC systems biology **1**, 28 (2007). DOI 10.1186/1752-0509-1-28. URL http://www.ncbi.nlm.nih.gov/pubmed/17567903http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC1904462
11. M. Tomita, K. Hashimoto, K. Takahashi, T. Shimizu, Y. Matsuzaki, F. Miyoshi, K. Saito, S. Tanida, K. Yugi, J. Venter, C. Hutchison, Bioinformatics **15**(1), 72 (1999). DOI 10.1093/bioinformatics/15.1.72. URL https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/15.1.72
12. C. Angione, P. Lió, Scientific Reports **5**(1), 15147 (2015). DOI 10.1038/srep15147. URL http://www.nature.com/articles/srep15147
13. A. Ebrahim, J.A. Lerman, B.O. Palsson, D.R. Hyduke, BMC Systems Biology **7**(1), 74 (2013). DOI 10.1186/1752-0509-7-74. URL http://bmcsystbiol.biomedcentral.com/articles/10.1186/1752-0509-7-74
14. J.A. Gomez, P.I. Barton, pp. 353–370 (2017). DOI 10.1007/978-1-4939-7528-0_16. URL https://doi.org/10.1007/978-1-4939-7528-0_16
15. G. Sierksma, *Linear and Integer Programming: Theory and Practice, Second Edition*. Advances in Applied Mathematics (Taylor & Francis, 2001). URL https://books.google.it/books?id=jP6H4xrTBkUC
16. C. Angione, N. Pratanwanich, P. Lió, ACS Synthetic Biology **4**(8), 880 (2015). DOI 10.1021/sb5003407. URL http://pubs.acs.org/doi/abs/10.1021/sb5003407
17. C. Angione, Bioinformatics **34**(3), 494 (2017). DOI 10.1093/bioinformatics/btx562. URL https://doi.org/10.1093/bioinformatics/btx562
18. E.L. Kaplan, P. Meier, Journal of the American Statistical Association **53**(282), 457 (1958). DOI 10.1080/01621459.1958.10501452. URL http://www.tandfonline.com/doi/abs/10.1080/01621459.1958.10501452
19. A. Iuliano, A. Occhipinti, C. Angelini, I. De Feis, P. Lió, Frontiers in physiology **7**, 208 (2016). DOI 10.3389/fphys.2016.00208. URL http://www.ncbi.nlm.nih.gov/pubmed/27378931http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4911360
20. W. Zhang, T. Ota, V. Shridhar, J. Chien, B. Wu, R. Kuang, PLoS Computational Biology **9**(3), e1002975 (2013). DOI 10.1371/journal.pcbi.1002975. URL http://dx.plos.org/10.1371/journal.pcbi.1002975
21. C. Li, H. Li, The Annals of Applied Statistics **4**(3), 1498 (2010). DOI 10.1214/10-AOAS332. URL https://arxiv.org/pdf/1011.3360.pdf
22. E.W. Weisstein. Laplacian Matrix. URL http://mathworld.wolfram.com/LaplacianMatrix.html
23. S.P. Boyd, L. Vandenberghe, *Convex optimization* (Cambridge University Press, 2004). URL http://www.cambridge.org/gb/academic/subjects/statistics-probability/optimization-or-and-risk/convex-optimization?format=HB#uw2F2Bu5ORJAmE8i.97
24. S. Diamond, S. Boyd, Journal of Machine Learning Research **17**, 1 (2016). URL http://web.stanford.edu/~boyd/papers/pdf/cvxpy_paper.pdf
25. I. Thiele, T.D. Vo, N.D. Price, B.O. Palsson, Journal of Bacteriology **187**(16), 5818 (2005). DOI 10.1128/JB.187.16.5818-5830.2005. URL http://www.ncbi.nlm.nih.gov/pubmed/16077130http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC1196094http://jb.asm.org/cgi/doi/10.1128/JB.187.16.5818-5830.2005
26. Z.A. King, J. Lu, A. Dräger, P. Miller, S. Federowicz, J.A. Lerman, A. Ebrahim, B.O. Palsson, N.E. Lewis, Nucleic Acids Research **44**(D1), D515 (2016). DOI 10.1093/nar/gkv1049. URL https://academic.oup.com/nar/article-lookup/doi/10.1093/nar/gkv1049
27. G. Sambamoorthy, K. Raman, Bioinformatics **34**(17), i981 (2018). DOI 10.1093/bioinformatics/bty604. URL https://doi.org/10.1093/bioinformatics/bty604

6 Code Guide

The code is written in such a way as to enable the user to avoid as much of the underlying complexity as possible. A guide to producing results similar to those listed in this paper follows;

```
import cobra
import pareto
import pa_re
import ne_re
```

Cobra, the pareto, pareto reconstruction and network regression files are imported. Next the model is loaded and the objectives are chosen. The example listed is for the *E. coli* model found on BiGG models.

```
model_str = 'iJO1366.json'
model = cobra.io.load_json_model(model_str)
obj1_str = ' BIOMASS_Ec_iJO1366_core_53p95M'
obj1 = model.reactions.get_by_id(obj1_str).flux_expression
obj2_str = 'EX_O2_e'
obj2 = model.reactions.get_by_id(obj2_str).flux_expression
filename = 'test_data.json'
```

Basic parameters for the dataset generation, regression and reconstruction can then be set.

```
LAMBD = 1.0
ALPHA = 0.5
CUTOFF = 99
GENS = 20
INDIV = 200
NODES = 40
RECON_POINTS = 200
CORES = 0
```

Here, the values of ($\lambda = 1.0, \alpha = 0.5, \tau = 0.99$) are used and typical values for the number of generations, individuals per generation and reconstruction points. This would use the single threaded version of the code, but 'CORES' can be set to the desired number of cores to be used in order to enable multithreading (with 0 using the sequential version).

Running Pareto data generation and storing to initial JSON file.

The initial Pareto data can then be generated. This data is converted into a dictionary which can then be easily stored within a JSON file.

```
pops, vals, pareto_data = pareto.pareto(GENS, INDIV, model, obj1, obj2,cores = CORES)
to_save = {'obj1_str': str(obj1), 'obj2_str': str(obj2), 'model' : model_str,'pareto': [{'
    obj1': p.fitness.values[0], 'obj2':p.fitness.values[1],'gene_set': list(p)} for p in
    pareto_data]}
with open(filename, 'w') as outfile:
  json.dump(to_save, outfile)
```

Running the network regression and storing in JSON file is simply done with a single command.

```
ne_re.add_linear_regression(filename, CUTOFF)
```

The Pareto Reconstruction can then be added to the JSON file.

```
pareto_left,pareto_right,pareto_noise,pareto_y,pareto_x = pa_re.reconstruct(filename, NODES,
    RECON_POINTS, model, obj1, obj2,cores=CORES)
```

These individual parts are all stored within the JSON file, so can then be easily plotted at a later date, and a batch script can easily be used to generate large amounts of complete data.