Aalto University

OPEN ACCESS

This is an electronic reprint of the original article.
This reprint may differ from the original in pagination and typographic detail.

Jokinen, Emmi; Heinonen, Markus; Lähdesmäki, Harri

MGPfusion

Published in:
Bioinformatics

DOI:
10.1093/bioinformatics/bty238

Published: 01/07/2018

Document Version
Publisher's PDF, also known as Version of record

Published under the following license:
CC BY-NC

Please cite the original version:
Jokinen, E., Heinonen, M., & Lähdesmäki, H. (2018). MGPfusion: Predicting protein stability changes with Gaussian process kernel learning and data fusion. *Bioinformatics*, *34*(13), i274-i283. https://doi.org/10.1093/bioinformatics/bty238

This material is protected by copyright and other intellectual property rights, and duplication or sale of all or part of any of the repository collections is not permitted, except that material may be duplicated by you for your research use or educational purposes in electronic or print form. You must obtain permission for any other use. Electronic or print copies may not be offered, whether for sale or otherwise to anyone who is not an authorised user.

Bioinformatics, 34, 2018, i274–i283
doi: 10.1093/bioinformatics/bty238
ISMB 2018

OXFORD

mGPfusion: predicting protein stability changes with Gaussian process kernel learning and data fusion

Emmi Jokinen[1,*], Markus Heinonen[1,2] and Harri Lähdesmäki[1]

[1]Department of Computer Science, Aalto University, Espoo 02150, Finland and [2]Helsinki Institute for Information Technology, Espoo 02150, Finland

*To whom correspondence should be addressed.

Abstract

Motivation: Proteins are commonly used by biochemical industry for numerous processes. Refining these proteins' properties via mutations causes stability effects as well. Accurate computational method to predict how mutations affect protein stability is necessary to facilitate efficient protein design. However, accuracy of predictive models is ultimately constrained by the limited availability of experimental data.

Results: We have developed mGPfusion, a novel Gaussian process (GP) method for predicting protein's stability changes upon single and multiple mutations. This method complements the limited experimental data with large amounts of molecular simulation data. We introduce a Bayesian data fusion model that re-calibrates the experimental and *in silico* data sources and then learns a predictive GP model from the combined data. Our protein-specific model requires experimental data only regarding the protein of interest and performs well even with few experimental measurements. The mGPfusion models proteins by contact maps and infers the stability effects caused by mutations with a mixture of graph kernels. Our results show that mGPfusion outperforms state-of-the-art methods in predicting protein stability on a dataset of 15 different proteins and that incorporating molecular simulation data improves the model learning and prediction accuracy.

Availability and implementation: Software implementation and datasets are available at github.com/emmijokinen/mgpfusion.

Contact: emmi.jokinen@aalto.fi

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Proteins are used in various applications by pharmaceutical, food, fuel and many other industries and their usage is growing steadily (Kirk *et al.*, 2002; Sanchez and Demain, 2011). Proteins have important advantages over chemical catalysts, as they are derived from renewable resources, are biodegradable and are often highly selective (Cherry and Fidantsef, 2003). Protein engineering is used to further improve the properties of proteins, for example to enhance their catalytic activity, modify their substrate specificity or to improve their thermostability (Rapley and Walker, 2000). Increasing the stability is an important aspect of protein engineering, as the proteins used in industry should be stable in the industrial process conditions, which often involve higher than ambient temperature and non-aqueous solvents (Bommarius *et al.*, 2011). The properties of a protein are modified by introducing alterations to its amino acid sequence. Mutations in general tend to be destabilising, and if too many destabilising mutations are implemented, the protein may not remain functional without compensatory stabilising mutations (Tokuriki and Tawfik, 2009).

The stability of a protein can be defined as the difference in Gibbs energy ΔG between the folded and unfolded (or native and denaturated) state of the protein. More precisely, the Gibbs energy difference determines the thermodynamic stability ΔG_t of the protein, as it does not take into account the kinetic stability ΔG_k which determines the energy needed for the transition between the folded and unfolded states (Anslyn and Dougherty, 2006) (see Supplementary Fig. S1). Here we will consider only the thermodynamic stability and from now on it will be referred to merely as stability ΔG.

The effect of mutations can be defined by the change they cause to the Gibbs energy ΔG, denoted as $\Delta\Delta G$ (Pace and Scholtz, 1997). To comprehend the significance of stability changes upon mutations,

© The Author(s) 2018. Published by Oxford University Press.

This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (http://creativecommons.org/licenses/by-nc/4.0/), which permits non-commercial re-use, distribution, and reproduction in any medium, provided the original work is properly cited. For commercial re-use, please contact journals.permissions@oup.com

Downloaded from https://academic.oup.com/bioinformatics/article-abstract/34/13/i274/5045756
by Helsinki University of Technology user
on 14 August 2018

we can consider globular proteins, the most common type of enzymes, whose polypeptide chain is folded up in a compact ball-like shape with an irregular surface (Alberts *et al.*, 2007). These proteins are only marginally stable and the difference in Gibbs energy between the folded and unfolded state is only about 5–15 kcal/mol, which is not much more than the energy of a single hydrogen bond that is about 2–5 kcal/mol (Branden and Tooze, 1999). Therefore, even one mutation that breaks a hydrogen bond can prevent a protein from folding properly.

The protein stability can be measured with many techniques, including thermal, urea and guanidinium chloride (GdmCl) denaturation curves that are determined as the fraction of unfolded proteins at different temperatures or at different concentrations of urea or GdmCl (Pace and Shaw, 2000). Some of the experimentally measured stability changes upon mutations have been gathered in thermodynamic databases such as Protherm (Kumar *et al.*, 2006).

A variety of computational methods have been introduced to predict the stability changes upon mutations more effortlessly than through experimental measurements. These methods utilize physics or knowledge-based potentials (Leaver-Fay *et al.*, 2011), their combinations, or different machine learning methods. The machine learning methods utilize support vector machines (SVM) (Capriotti *et al.*, 2005b, 2008; Chen *et al.*, 2013; Cheng *et al.*, 2005; Folkman *et al.*, 2014; Liu and Kang, 2012; Pires *et al.*, 2014a), random forests (Tian *et al.*, 2010; Wainreb *et al.*, 2011), neural networks (Dehouck *et al.*, 2009; Giollo *et al.*, 2014) and Gaussian processes (Pires *et al.*, 2014b). However, it has been assessed that although on average many of these methods provide good results, they tend to fail on details (Potapov *et al.*, 2009). In addition, many of these methods are able to predict the stability effects only for single-point mutations.

We introduce mGPfusion (mutation Gaussian Processes with data fusion), a method for predicting stability effects of both point and multiple mutations. mGPfusion is a protein-specific model—in contrast to earlier stability predictors that aim to estimate arbitrary protein structure or sequence stabilities—and achieves markedly higher accuracy while utilising data only from a single protein at a time. In contrast to earlier works that only use experimental data to train their models, we also combine exhaustive Rosetta (Leaver-Fay *et al.*, 2011) simulated point mutation *in silico* stabilities to our training data.

A key part of mGPfusion is the automatic scaling of simulated data to better match the experimental data distribution based on those variants that have both experimental and simulated stability values. Furthermore, we estimate a variance resulting from the scaling, which places a higher uncertainty on very destabilising simulations. Our Gaussian process model then utilizes the joint dataset with their estimated heteroscedastic variances and uses a mixture of graph kernels to assess the stability effects caused by changes in amino acid sequence according to 21 substitution models. Our experiments on a novel 15 protein dataset show a state-of-the-art stability prediction performance, which is also sustained when there is access only to a very few experimental stability measurements.

2 Materials and methods

Following Pires *et al.* (2014b) we choose a Bayesian model family of Gaussian processes for prediction of mutation effects on protein stability due to its inherent ability to handle uncertainty in a principle way. Bayesian modelling is a natural approach for combining the experimental and simulated data distribution, while it is also suitable for learning the underlying mixture of substitution models that governs the mutational process.

The pipeline for mGPfusion is presented in Figure 1. The first part of mGPfusion consists of collection of *in silico* and experimental datasets discussed in Section 2.1, the scaling of the *in silico* dataset in Section 2.2 and the fusion of these two datasets in Section 2.3. The second part consists of the Gaussian process model described in Section 2.4 with detailed description of the graph kernels in Sections 2.5–2.6 and model inference in Section 2.7. Finally, the evaluation criteria used are described in Section 2.8.

2.1 Experimental and *in silico* data

Protherm is a database of numerical thermodynamic parameters for proteins and their mutants (Kumar *et al.*, 2006). From Protherm we gathered all proteins with at least 50 unique mutations whose $\Delta\Delta G$ has been measured by thermal denaturation, and where a PDB code for a 3D structure of the protein was reported. We required the proteins to have at least 50 unique mutations, so that we would have a representative test set and get sufficiently reliable estimates of prediction accuracy on individual proteins and examine how the amount of experimental training data affects the accuracy of the model. The 3D structures are necessary for obtaining the connections between residues. We collected the 3D structures with the reported PDB codes from the Protein Databank, www.rcsb.org (Berman *et al.*, 2000). The 15 proteins that fulfilled these requirements are listed in Table 1. We averaged the stability values for proteins with multiple measurements and ignored mutations to residues not present in their 3D structures. These datasets are available at github.com/emmijokinen/mgpfusion.

We also generate simulated data of the stability effects of all possible single mutations of the proteins. Our method can utilize any simulated stability values. We used the 'ddG monomer' application of Rosetta 3.6 (Leaver-Fay *et al.*, 2011) using the high-resolution backrub-based protocol 16 recommended in Kellogg *et al.* (2011). The predictions $\mathbf{y}^S$ made with Rosetta are given in Rosetta Energy Units (REU). Kellogg *et al.* (2011) suggest transformation $0.57y^S$ for converting the predictions into physical units. The simulated data scaled this way is not as accurate as the experimental data, the correlation and root mean square error (rmse) with respect to the experimental data are shown for all proteins in Table 2 and for individual proteins in Supplementary Table S2, on rows labelled Rosetta. For this reason, we use instead a Bayesian scaling described in the next section and different noise models for the experimental and simulated data, described in Section 2.3.

For each of the 15 proteins, let $\mathbf{x}_i = (x_{i1}, \ldots, x_{iM})$ denote its M-length variant i with positions p labelled with residues $x_{ip} \in \{A, R, N, \ldots, V\}$. We denote the wild-type protein as $\mathbf{x}_0$. We collect 15 separate sets of simulated and experimental data. We denote the N_E experimental variants of each protein as $X_E = (\mathbf{x}_1^E, \ldots, \mathbf{x}_{N_E}^E)^T$ with the corresponding experimental stability values $\mathbf{y}_E = (y_1^E, \ldots, y_{N_E}^E)^T \in \mathbb{R}^{N_E}$. Similarly, we denote the N_S simulated observations as $X_S = (\mathbf{x}_1^S, \ldots, \mathbf{x}_{N_S}^S)^T$ and $\mathbf{y}_S = (y_1^S, \ldots, y_{N_S}^S)^T \in \mathbb{R}^{N_S}$.

2.2 Bayesian scaling of *in silico* data

The described transformation from REU to physical units may not be optimal for all proteins. We therefore applied instead a linear-exponential scaling function to obtain *scaled* Rosetta simulated stabilities $\tilde{y}^S$,

Downloaded from https://academic.oup.com/bioinformatics/article-abstract/34/13/i274/5045756
by Helsinki University of Technology user
on 14 August 2018

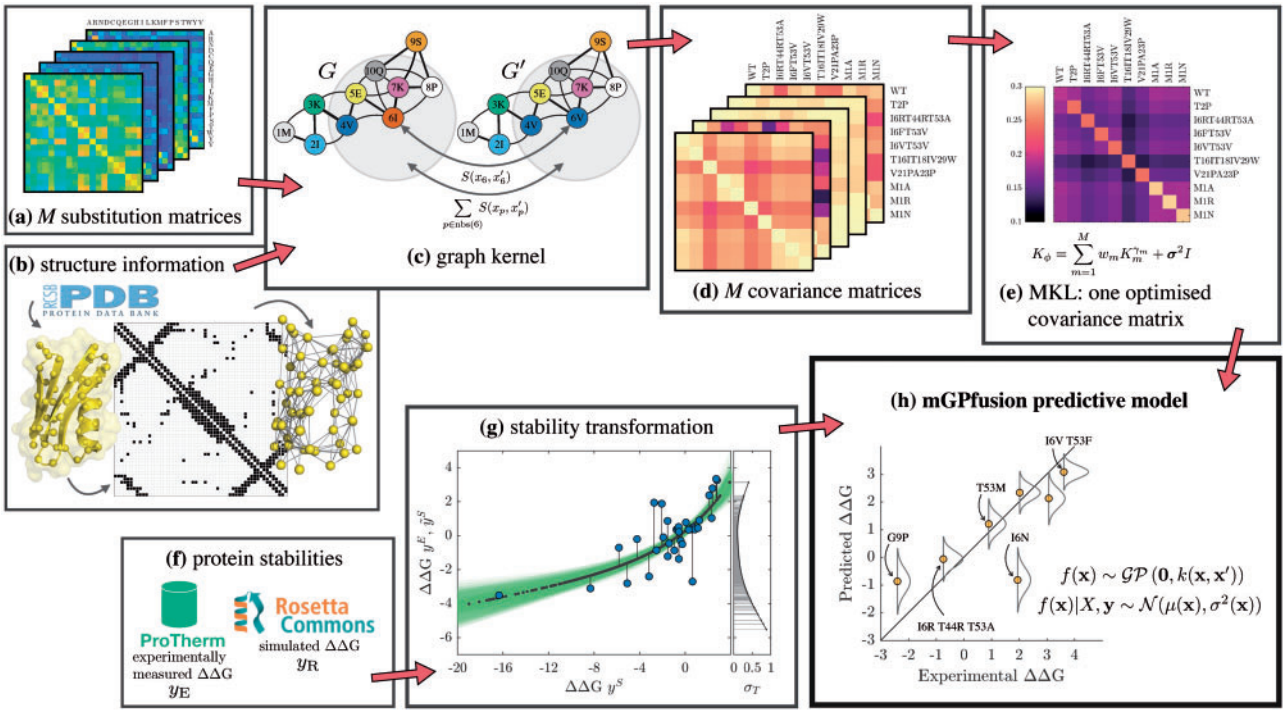


Fig. 1. Pipeline illustration for mGPfusion. (**a**) $M=21$ substitution matrices utilize different information sources and give scores to pairwise amino acid substitutions. (**b**) The wild-type structures from Protein Data Bank are modelled as contact graphs. (**c**) The graph kernel measures similarity of two sequences by a substitution model S over all positions p and their neighbourhoods nbs(p) in the contact graph. (**d**) Each substitution matrix is used to create a separate covariance matrix. (**e**) Multiple kernel learning (MKL) is used for finding the optimal combination of the base kernels. The kernel matrix measures variant similarities. (**f**) Experimentally measured ΔΔG values y^E are gathered from Protherm and Rosetta's ddg monomer application is used to simulate the stability effects y^S for all single point mutations. (**g**) Bayesian scaling for the simulated values y^S at the x-axis. Possible scalings are coloured with green and the chosen scaling from y^S into scaled values $\tilde{y}^S$ is marked by black dots. The scaling is fitted to a subset of experimentally measured stabilities y^E (circles). (**h**) The stability predictive GP model is trained using experimental and simulated data through the kernel matrix

$$\tilde{y}^S = g(y^S \mid \theta_j) = a_j e^{c_j y^S} + b_j y^S + d_j. \tag{1}$$

This scaling transforms the Rosetta simulations y^S for each protein $j = 1, \ldots, 15$ to correspond better to the experimental data. The parameters $\theta_j = (a_j, b_j, c_j, d_j)$ define the weight a_j and steepness c_j of the exponential term, while the linear term has slope b_j and intercept d_j. To avoid overfitting, we perform Bayesian linear regression and start by defining parameter prior $p(\theta_j) = p(a_j)p(b_j)p(c_j)p(d_j)$ that reflects our beliefs about realistic scalings having only moderate steepness:

$$\begin{aligned} p(a_j) &= \text{Gamma}(a_j \mid \alpha_a, \beta_a) \\ p(b_j) &= \text{Beta}(1/2 \cdot b_j \mid \alpha_b, \beta_b) \\ p(c_j) &= \text{Beta}(10/3 \cdot c_j \mid \alpha_c, \beta_c) \\ p(d_j) &= \mathcal{N}(d_j \mid \mu_d, \sigma_d^2). \end{aligned} \tag{2}$$

The empirically selected hyperparameter values are listed in Supplementary Table S1 and the priors are illustrated in Supplementary Figure S2.

We compute the posterior for θ_j using the subset of simulated data that have corresponding experimentally measured data:

$$p(\theta_j|\mathbf{y}_E, \mathbf{y}_S) \propto \prod_{i:\mathbf{x}_i \in X_E \cap X_S} \mathcal{N}(y_i^E \mid g(y_i^S|\theta_j), \sigma_n^2) p(\theta_j).$$

The product iterates over all $N_{E\cap S}$ simulated ΔΔG's that have a matching experimentally observed value. The σ_n^2 is the scaling error variance, which was set to $\sigma_n^2 = 0.5$. The parameters θ for each protein were sampled using a random walk Metropolis-Hastings MCMC algorithm (the mhsample function in Matlab) for $N_{MC} = 10\,000$ samples with a burn-in set to 500. The proposal distribution was selected to be a symmetric uniform distribution such that $[a_{s+1}, b_{s+1}, c_{s+1}, d_{s+1}] \sim U(a_s \pm 0.4, b_s \pm 0.04, c_s \pm 0.04, d_s \pm 0.4)$. Given the sample of scaling parameters $(\theta_j^{(s)})_{s=1}^{N_{MC}}$, we define the scaled simulated data as the average scaling over the MCMC sample, and record also the sample scaling variance

$$\tilde{y}_i^S = \frac{1}{N_{MC}} \sum_{s=1}^{N_{MC}} g(y_i^S|\theta_j^{(s)}) \tag{3}$$

$$\sigma_T^2(i) = \frac{1}{N_{MC}} \sum_{s=1}^{N_{MC}} \left(g(y_i^S|\theta_j^{(s)}) - \tilde{y}_i^S \right)^2. \tag{4}$$

See Figure 1g for an illustration of the scaling. We collect the scaled simulated value and its variance from each simulated point into vectors $\tilde{\mathbf{y}}_S = (\tilde{y}_1^S, \ldots, \tilde{y}_{N_S}^S)$ and $\boldsymbol{\sigma}_T^2 = (\sigma_T^2(1), \ldots, \sigma_T^2(N_S)) \in \mathbb{R}^{N_S}$.

2.3 Data fusion and noise models

For each protein j, we organize its experimental data $(X_E, \mathbf{y}_E)$ and transformed simulated data $(X_S, \tilde{\mathbf{y}}_S)$ along with the wild-type information $(\mathbf{x}_0, y_0)$ into a single joint dataset of variants $X = (\mathbf{x}_0, X_E, X_S)$ and stabilities $\mathbf{y} = (y_0, \mathbf{y}_E, \tilde{\mathbf{y}}_S)$ of size $\mathbb{R}^N$ where $N = 1 + N_E + N_S$ is the total number of simulated and experimental data points, including the wild-type. We assume heteroscedastic additive noise models for the three information sources

Downloaded from https://academic.oup.com/bioinformatics/article-abstract/34/13/i274/5045756
by Helsinki University of Technology user
on 14 August 2018

Table 1. The 15 protein data from ProTherm database with counts of point mutations, all mutations and of simulated point mutation stability changes

Protein (organism)	PDB	Mutations		
		all	point	point (sim)
T4 Lysozyme (Enterobacteria phage T4)	2LZM	349	264	3116
Barnase (*Bacillus amyloliquefaciens*)	1BNI	182	163	2052
Gene V protein (*Escherichia* virus m13)	1VQB	124	92	1634
Glycosyltransferase A (*Homo sapiens*)	1LZI	116	114	2470
Chymotrypsin inhibitor 2 (*Hordeum vulgare*)	2CI2	98	77	1235
Protein G (*Streptococcus* sp. gx7805)	1PGA	89	34	1064
Ribonuclease H (*Escheria coli*)	2RN2	83	65	2945
Cold shock protein B (*Bacillus subtilis*)	1CSP	80	50	1273
Apomyoglobin (*Physeter catodon*)	1BVC	80	56	2907
Hen egg white lysozyme (*Gallus gallus*)	4LYZ	63	50	2451
Ribonuclease A (*Bos taurus*)	1RTB	57	50	2356
Peptidyl-prolyl cis-trans isomerase (*Homo sapiens*)	1PIN	56	56	2907
Ribonuclease T1 isozyme (*Aspergillus oryzae*)	1RN1	53	48	1957
Ribonuclease (*Streptomyces auerofaciens*)	1RGG	54	45	1824
Bovine pancreatic trypsin inhibitor (*Bos taurus*)	1BPI	53	47	1102
Total		1537	1211	31293

Table 2. Comparison of different methods on the 15 protein dataset with respect to ρ and rmse

Method	Correlation ρ									rmse								
	Point mutations			Multiple mutations			All mutations			Point mutations			Multiple mutations			All mutations		
	Cross-validation level			Cross-validation level			Cross-validation level			Cross-validation level			Cross-validation level			Cross-validation level		
	mut.	pos.	prot.	mut.	pos.	prot.	mut.	pos.	prot.	mut.	pos.	prot.	mut.	pos.	prot.	mut.	pos.	prot.
mGPfusion	**0.81**	**0.70**	**0.56**	**0.88**	0.61	0.49	**0.83**	0.64	0.52	1.07	**1.26**	**1.61**	**1.33**	2.45	2.53	**1.13**	1.87	1.84
mGPfusion, only B62	0.79	0.69	0.56	0.86	**0.64**	**0.50**	0.82	**0.66**	**0.52**	1.11	1.30	1.62	1.43	**2.40**	**2.50**	1.18	**1.85**	**1.84**
mGP	0.81	0.51	–	0.86	0.52	–	0.83	0.50	–	**1.04**	1.54	–	1.44	2.65	–	1.14	2.09	–
mGP, only B62	0.76	0.34	–	0.86	0.55	–	0.80	0.49	–	1.26	1.95	–	1.45	2.56	–	1.30	2.23	–
Rosetta scaled	0.65	0.63	–	0.51	0.39	–	0.60	0.48	–	1.35	1.38	–	2.49	2.99	–	1.66	2.22	–
Predictions from off-the-shelf implementations with no cross-validation																		
Rosetta	0.55			0.40			0.49			1.63			2.74			1.92		
mCSM	0.61			–			–			1.40			–			–		
PoPMuSiC	0.64			–			–			1.37			–			–		

Note: Mutation, position and protein are referred to as mut., pos. and prot., respectively. Largest correlation value or smallest rmse of each column is bolded for convenience. Predictions from off-the-shelf implementations of Rosetta, mCSM and PoPMuSiC are used directly without cross-validation.

$$\begin{aligned} y_0 &= f(\mathbf{x}_0) + \varepsilon_0,\ \varepsilon_0 \sim \mathcal{N}(0, \sigma_0^2) \\ y_i^E &= f(\mathbf{x}_i^E) + \varepsilon_i^E,\ \varepsilon_i^E \sim \mathcal{N}(0, \sigma_E^2) \\ \tilde{y_i^S} &= f(\mathbf{x}_i^S) + \varepsilon_i^S,\ \varepsilon_i^S \sim \mathcal{N}(0, (\sigma_E + \sigma_S + t\sigma_T(i))^2), \end{aligned} \tag{5}$$

where the observed values are noisy versions of the underlying 'true' stability function $f(\mathbf{x})$ corrupted by zero-mean noise with data source specific variances. We learn a Gaussian process based stability function $f(\mathbf{x})$ in the next Section.

The Equations (5) encode that the experimental data are corrupted by a global experimental noise variance σ_E^2. The simulated stabilities are additionally corrupted by a global Rosetta simulator error variance σ_S^2, and by the value-dependent transformation variance $t\sigma_T^2(i)$ scaled by parameter t. The model then encapsulates that we trust the Rosetta data less than the experimental data. By definition, the ΔΔG of the wild-type is zero ($y_0 = 0$) with very small assumed error, $\sigma_0 = 10^{-6}$. Note that $\boldsymbol{\sigma}_T^2$ are fixed by Equation (4), while we infer the optimal values for the remaining three free parameters (σ_E, σ_R, t) (see Section 2.4). The parameters σ_E^2 and σ_S^2 are assigned priors

$$\begin{aligned} \sigma_E &\sim \text{Gamma}(\sigma_E | \alpha_E, \beta_E) \\ \sigma_S &\sim \text{Gamma}(\sigma_R | \alpha_S, \beta_S). \end{aligned} \tag{6}$$

The values of these hyperparameters are shown in Supplementary Table S1.

2.4 Gaussian processes

We use a Gaussian process (GP) function f to predict the stability $f(\mathbf{x}) \in \mathbb{R}$ of a protein variant $\mathbf{x}$. Gaussian processes are a family of non-parametric, non-linear Bayesian models (Rasmussen and Williams, 2006). A zero-mean GP prior

$$f(\mathbf{x}) \sim \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}')),$$

defines a distribution over functions $f(\mathbf{x})$ whose mean and covariance are

$$\mathbb{E}[f(\mathbf{x})] = 0$$

$$\mathbf{cov}[f(\mathbf{x}), f(\mathbf{x}')] = k(\mathbf{x}, \mathbf{x}').$$

Downloaded from https://academic.oup.com/bioinformatics/article-abstract/34/13/i274/5045756
by Helsinki University of Technology user
on 14 August 2018

For any collection of protein variants $X = \mathbf{x}_1, \ldots, \mathbf{x}_N$, the function values follow a multivariate normal distribution $\mathbf{f} \sim \mathcal{N}(0, K_{XX})$, where $\mathbf{f} = (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_N))^T \in \mathbb{R}^N$, and where $K_{XX} \in \mathbb{R}^{N \times N}$ with $[K_{XX}]_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. The key property of Gaussian processes is that they encode functions that predict similar stability values $f(\mathbf{x}), f(\mathbf{x}')$ for protein variants $\mathbf{x}, \mathbf{x}'$ that are similar, as encoded by the kernel $k(\mathbf{x}, \mathbf{x}')$. The key part of GP modelling is then to infer a kernel that measures the mutation's effects to the stability.

Let a dataset of noisy stability values from two sources be $\mathbf{y} \in \mathbb{R}^N$, the corresponding protein structures $X = (\mathbf{x}_i)_{i=1}^N$, and a new protein variant $\mathbf{x}_*$ whose stability we wish to predict. A Gaussian process defines a joint distribution over the observed values $\mathbf{y}$ of variants X, and the unknown function value $f(\mathbf{x}_*)$ of the unseen variant $\mathbf{x}_*$,

$$\begin{bmatrix} \mathbf{y} \\ f(\mathbf{x}_*) \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K_{XX} + \mathrm{diag}(\boldsymbol{\sigma}^2) & \mathbf{k}_{X*} \\ \mathbf{k}_{*X} & k(\mathbf{x}_*, \mathbf{x}_*) \end{bmatrix}\right),$$

where $\mathbf{k}_{X*} = \mathbf{k}_{*X}^T \in \mathbb{R}^N$ is a kernel vector with elements $k(\mathbf{x}_i, \mathbf{x}_*)$ for all $i = 1, \ldots, N$, and where $\boldsymbol{\sigma}^2 = (\sigma_0^2, \sigma_E^2 \mathbf{1}^T, (\sigma_E \mathbf{1}^T + \sigma_S \mathbf{1}^T + t\boldsymbol{\sigma}_T{}^T)^2)^T$ collects final variances of the data points from Equations (5). Here the exponents are elementwise. The conditional distribution gives the posterior distribution of the stability prediction as

$$f(\mathbf{x}_*)|(X, \mathbf{y}) \sim \mathcal{N}(\mu(\mathbf{x}_*), \sigma^2(\mathbf{x}_*)),$$

where the prediction mean and variance are

$$\mu(\mathbf{x}_*) = \mathbf{k}_{*X}(K_{XX} + \mathrm{diag}(\boldsymbol{\sigma}^2))^{-1}\mathbf{y},$$
$$\sigma^2(\mathbf{x}_*) = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_{*X}(K_{XX} + \mathrm{diag}(\boldsymbol{\sigma}^2))^{-1}\mathbf{k}_{X*}.$$

Hence, in GP regression the stability predictions $\mu(\mathbf{x}_*) \pm \sigma(\mathbf{x}_*)$ will come with uncertainty estimates.

2.5 Graph kernel

Next, we consider how to compute the similarity function $k(\mathbf{x}, \mathbf{x}')$ between two variants of the same protein structure. We will encode the 3D structural information of the two protein variants as a contact map and measure their similarity by the formalism of graph kernels (Vishwanathan *et al.*, 2010).

We consider two residues to be in contact if their closest atoms are within 5 Å of each other in the PDB structure, which is illustrated in Figure 1b. All variants of the same protein have the same length, with only different residues at mutating positions. Furthermore, we assume that all variants share the wild-type protein contact map.

To compare protein variants, we construct a weighted decomposition kernel (WDK) (Menchetti *et al.*, 2005) between two protein variants $\mathbf{x} = (x_1, \ldots, x_M)$ and $\mathbf{x}' = (x'_1, \ldots, x'_M)$ of length M,

$$k(\mathbf{x}, \mathbf{x}') = \sum_{p=1}^{M} \left(S(x_p, x'_p) \sum_{l \in \mathrm{nbs}(p)} S(x_l, x'_l) \right), \quad (7)$$

where $\mathrm{nbs}(p)$ defines the set of neighbouring positions to position p, and S is a substitution matrix. The kernel iterates over all positions p and compares for each of them their residues through a substitution matrix $S(x_p, x'_p)$. Furthermore, the similarity of the residues at each position is multiplied by the average similarity of the residues at its neighbouring positions $S(x_l, x'_l)$. Hence, the kernel defines the similarity of two protein variants as the average position and neighbourhood similarity over all positions. The kernel matrix is normalized so that for two data points, $\mathbf{x}$ and $\mathbf{x}'$, the normalized kernel is $\widehat{k}(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{x}')/\sqrt{k(\mathbf{x}, \mathbf{x})k(\mathbf{x}', \mathbf{x}')}$, as defined by Shawe-Taylor and Cristianini (2004). The kernel is illustrated in Figure 1c.

The above WDK kernel allows us to compare the effects of multiple simultaneous mutations. However, as the wild type protein structure is used for all of the protein variants, changes that the mutations may cause to the protein structure are not taken into consideration. This may cause problems if mutations that alter the protein structure significantly are introduced—especially if many of them are introduced simultaneously. On the other hand, substitution matrices that have their basis in sequence comparisons, should take these effects into account to some extend as these kinds of mutations are usually highly destabilising and do not occur often in nature. In the next section, we will discuss how we utilize different substitution matrices with multiple kernel learning.

2.6 Substitution matrices and multiple kernel learning

The BLOSUM substitution models have been a common choice for protein models (Giguere *et al.*, 2013), while mixtures of substitution models were proposed by Cichonska *et al.* (2017). BLOSUM matrices score amino acid substitutions by their appearances throughout evolution, as they compare the frequencies of different mutations in similar blocks of sequences (Henikoff and Henikoff, 1992). However, there are also different ways to score amino acids substitutions, such as chemical similarity and neighbourhood selectivity (Tomii and Kanehisa, 1996). When the stability effects of mutations are evaluated, the frequency of an amino acid substitution in nature may not be the most important factor.

To take into account different measures of similarity between amino acids, we employed a set of 21 amino acid substitution matrices gathered from AAindex2 (http://www.genome.jp/aaindex/) (Kawashima *et al.*, 2007). AAindex2 currently contains 94 substitution matrices. From these we selected those that had no gaps concerning substitutions between the 20 naturally occurring amino acids and scaled them between zero and one as

$$S = \frac{\mathrm{S}_0 - \min(\mathrm{S}_0) + 1}{\max(\mathrm{S}_0) - \min(\mathrm{S}_0) + 1}. \quad (8)$$

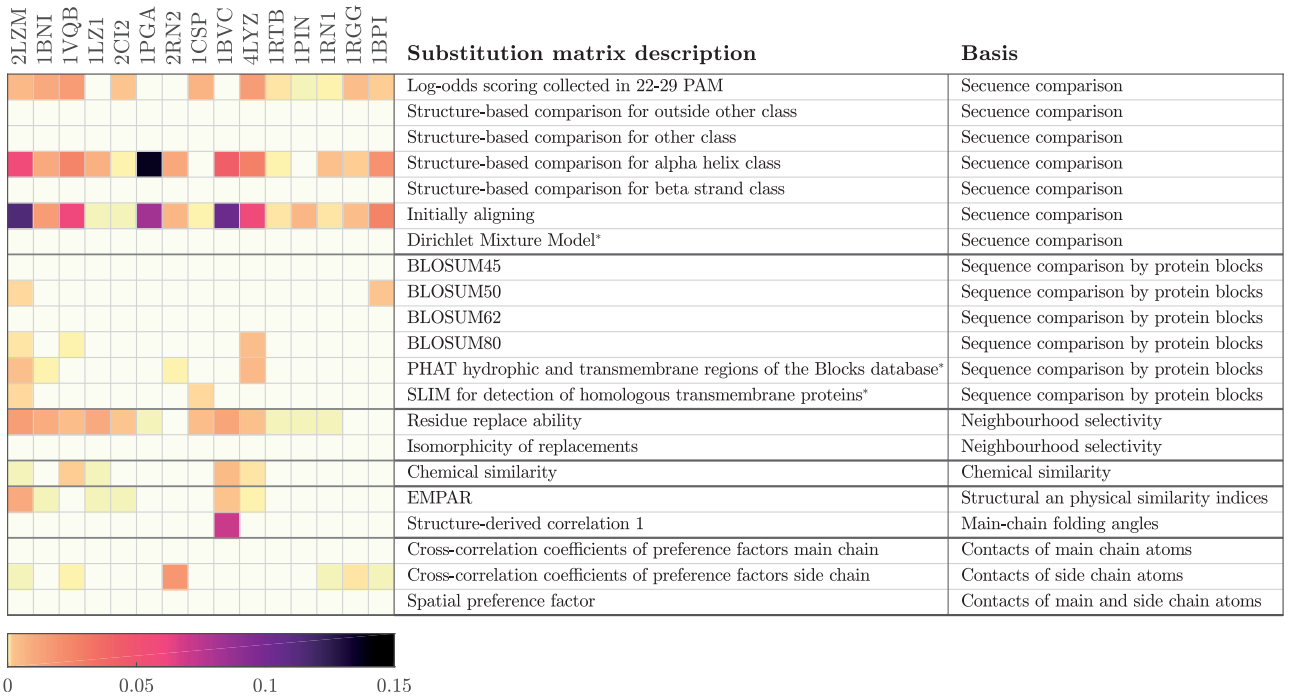
Out of these matrices, we only chose those 23 matrices that were positive semidefinite. Furthermore, there were two pairs of matrices that were extremely similar, and we only selected one matrix from each pair, ending up with 21 substitution matrices. These substitution matrices are used together with Equation 7 for computing 21 base kernel matrices. Finally, MKL is used to find an optimal combination of the base kernels of form

$$K_\phi = \sum_{m=1}^{21} w_m K_m^{(\gamma_m)}, \quad (9)$$

where w_m is a kernel specific weight, γ_m is an (elementwise) exponent. The elementwise exponent retains the SDP property of K_ϕ (Shawe-Taylor and Cristianini, 2004). We observe empirically that the optimal kernel weights w_m tend to be sparse (see Fig. 2).

The selected substitution matrices are listed in Figure 2. These matrices have different basis and through multiple kernel learning (MKL) our model learns which of these are important for inferring the stability effects that mutations cause on different proteins. The figure illustrates this by showing the average weights of the base kernel matrices obtained via the multiple kernel learning.

Downloaded from https://academic.oup.com/bioinformatics/article-abstract/34/13/i274/5045756
by Helsinki University of Technology user
on 14 August 2018



Substitution matrix description	Basis
Log-odds scoring collected in 22-29 PAM	Secuence comparison
Structure-based comparison for outside other class	Secuence comparison
Structure-based comparison for other class	Secuence comparison
Structure-based comparison for alpha helix class	Secuence comparison
Structure-based comparison for beta strand class	Secuence comparison
Initially aligning	Secuence comparison
Dirichlet Mixture Model*	Secuence comparison
BLOSUM45	Sequence comparison by protein blocks
BLOSUM50	Sequence comparison by protein blocks
BLOSUM62	Sequence comparison by protein blocks
BLOSUM80	Sequence comparison by protein blocks
PHAT hydrophic and transmembrane regions of the Blocks database*	Sequence comparison by protein blocks
SLIM for detection of homologous transmembrane proteins*	Sequence comparison by protein blocks
Residue replace ability	Neighbourhood selectivity
Isomorphicity of replacements	Neighbourhood selectivity
Chemical similarity	Chemical similarity
EMPAR	Structural an physical similarity indices
Structure-derived correlation 1	Main-chain folding angles
Cross-correlation coefficients of preference factors main chain	Contacts of main chain atoms
Cross-correlation coefficients of preference factors side chain	Contacts of side chain atoms
Spatial preference factor	Contacts of main and side chain atoms

Fig. 2. Average weights for kernels utilising the described substitution matrices from AAindex2, when GP models were trained with mutation level cross-validation. Basis for the substitution matrices are obtained from (Tomii and Kanehisa, 1996). * were added to AAindex2 in a later release, and their basis were not determined by Tomii and Kanehisa (1996)

2.7 Parameter inference

The complete model has five parameters $\phi = (\sigma_E, \sigma_S, t, \mathbf{w}, \boldsymbol{\gamma})$ to infer, of which the variance parameters (σ_E, σ_S, t) parameterise the joint data variance $\boldsymbol{\sigma}_\phi^2$, while the MKL parameters $\mathbf{w} = (w_1, \ldots, w_{21})$ and $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_{21})$ parameterise the kernel matrix K_ϕ. In a Gaussian process model these can be jointly optimized by the marginal (log) likelihood with priors

$$\begin{aligned}\log p(\mathbf{y}|\phi)p(\sigma_E)p(\sigma_R) &= \log \int p(\mathbf{y}|\mathbf{f},\phi)p(\mathbf{f}|\phi)p(\sigma_E)p(\sigma_R)d\mathbf{f} \\ &\propto -\frac{1}{2}\mathbf{y}^T\left(K_\phi + \mathrm{diag}\left(\boldsymbol{\sigma}_\phi^2\right)\right)^{-1}\mathbf{y} - \frac{1}{2}\log|K_\phi + \mathrm{diag}\left(\boldsymbol{\sigma}_\phi^2\right)| \\ &\quad + \log \mathrm{Gamma}(\sigma_E|\alpha_E,\beta_E) + \log \mathrm{Gamma}(\sigma_S|\alpha_S,\beta_S),\end{aligned} \tag{10}$$

which automatically balances model fit (the square term) and the model complexity (the determinant) to avoid overfitting (Rasmussen and Williams, 2006). The parameters can be optimized by maximising the marginal log likelihood (10) using gradient ascent, since the marginal likelihood can be differentiated analytically (see Supplementary Equations S1 and S2). We utilized a limited-memory projected quasi-Newton algorithm [minConf_TMP (http://www.cs.ubc.ca/~schmidtm/Software/minConf.html)], described by Schmidt *et al.* (2009).

2.8 Evaluation criteria

We chose to evaluate the accuracy of our predictions using the same metrics that have been used by many others—correlation ρ between the predicted and experimentally measured ΔΔG values (Capriotti *et al.*, 2005a; Dehouck *et al.*, 2009; Kellogg *et al.*, 2011; Pires *et al.*, 2014b; Potapov *et al.*, 2009) and the root mean square error (rmse) (Dehouck *et al.*, 2009; Pires *et al.*, 2014a,b), which are determined in the Supplementary Equations S3 and S4. We use marginal likelihood maximization to infer model parameters and perform cross-validation to evaluate the model performance on test data. Below we only report evaluation metrics obtained from the test sets not used at any stage of the model learning or data transformation sampling.

3 Results

In this section we evaluate the performance of mGPfusion on predicting stability effects of mutations, and compare it to the state-of-the-art prediction methods mCSM, PoPMuSiC and Rosetta. Rosetta is a molecular modelling software whose ddg_monomer module can directly simulate the stability changes ΔΔG of a protein upon mutations. PoPMuSic and mCSM are machine learning models that predict stability based on protein variant features. We run Rosetta locally, and use mCSM and PoPMuSiC models through their web servers (biosig.unimelb.edu.au/mcsm and omictools.com/popmusic-tool). This may give these methods an advantage over mGPfusion since parts of our testing data were likely used within their training data.

We compare four different variants of our method: mGPfusion that uses both simulated data and MKL, 'mGPfusion, only B62' that uses simulated data but incorporates only one kernel matrix (BLOSUM62 substitution matrix), mGP model that uses MKL but does not use simulated data, and 'mGP, only B62' that uses only the base GP model but does not incorporate simulated data and uses only the BLOSUM62 substitution matrix. In addition, we experiment on transforming Rosetta predictions with the Bayesian scaling. We perform the experiments for the 15 proteins separately using either position or mutation level (leave-one-out) cross-validation regarding the methods mGP, mGPfusion and the Bayesian scaling of Rosetta. Pires *et al.* (2014b) used protein and position level cross-validation to evaluate their model. In protein level cross-validation all mutations in a protein are either in the test or training set exclusively. When we train our model using protein level cross-validation, we use no experimental data and rely only on the

Downloaded from https://academic.oup.com/bioinformatics/article-abstract/34/13/i274/5045756
by Helsinki University of Technology user
on 14 August 2018

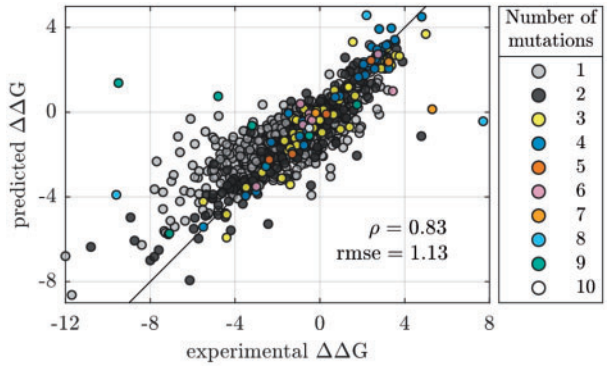


Fig. 3. Scatter plot for the mutation level (leave-one-out) predictions made for all 15 proteins (see Table 1). The colour indicates the number of simultaneous mutations

simulated data. Position level cross-validation is defined so that all mutations in a position are either in the test or training set exclusively. However, datasets in Pires *et al.* (2014b) contained only point mutations and therefore we had to extend the definition to also include multiple mutations. In position level cross-validation we train one model for each position using only the part of data that has a wild-type residue in that position. Therefore, in position level cross-validation we construct a test set that contains all protein variants that have a mutation at position p and use as training set all the protein variants that have a wild-type residue at that position. Dehouck *et al.* (2009) evaluated their models by randomly selecting training and test sets so that each mutation was exclusively in one of the sets, but both sets could contain mutations from the same position of the same protein. We call this mutation level cross-validation. When we use all available experimental data with mutation level cross-validation, this corresponds to leave-one-out cross-validation.

3.1 Predicting point mutations

Table 2 summarizes the average prediction performance over all 15 proteins for all compared methods, types of mutations and cross-validation types. We first compare the performances on single point mutations, where mGPfusion and mGP achieve the highest performance with $\rho = 0.81$ and rmse $= 1.07$ kcal/mol, and $\rho = 0.81$ and rmse $= 1.04$ kcal/mol, respectively with mutation level cross-validation. With only one kernel utilising the BLOSUM62 matrix instead of MKL, the performance decreases slightly, but the competing methods are still outperformed, as mCSM achieves $\rho = 0.64$ and rmse $= 1.37$ kcal/mol, PoPMuSic $\rho = 0.61$ and Rosetta $\rho = 0.55$. Applying Bayesian scaling on Rosetta simulator improves the performance of standard Rosetta from $\rho = 0.55$ to $\rho = 0.65$ and decreases the rmse from 1.63 to 1.35 kcal/mol, which is interestingly even slightly better than the performances of mCSCM and PoPMuSiC.

With position level cross-validation mGPfusion achieves the highest performance of $\rho = 0.70$ and rmse $= 1.26$ kcal/mol, likely due to having still access to simulated variants from that position, since they are always available to the learner. Without simulation data, the baseline machine learning model mGP performance decreases to $\rho = 0.51$ and rmse $= 1.54$ kcal/mol, thus demonstrating the importance of the data fusion. Cross-validation could not be performed for the off-the-shelf methods mCSM and PoPMuSiC. Even still, mGPfusion (trained with one or multiple kernels) outperforms competing state-of-the-art methods and achieves markedly higher prediction performance as quantified by both mutation and position level cross-validations. Also mGP outperforms these methods when quantified by mutation level cross-validation. With protein level cross-validation mGPfusion achieves slightly better results than Rosetta.

3.2 Predicting multiple mutations

Next, we tested stability prediction accuracies for variants containing either single or multiple mutations. Figure 3 shows a scatter plot of mGPfusion predictions for all 1537 single and multiple mutation variants (covering all 15 proteins) against the experimental ΔΔG values using the mutation level (leave-one-out) cross-validation. The points are coloured by the number of simultaneous mutations in the variants, with 326 variants having at least 2 mutations (see Table 1). Figure 3 illustrates the mGPfusion's overall high accuracy of $\rho = 0.83$ and rmse $= 1.13$ kcal/mol on both single and multiple mutations (see Table 2). Scatter plots for the individual proteins can be found in Supplementary Figure S3. Dehouck *et al.* (2009) suggested that considering the predictive power after removal of most badly predicted stability effects of mutations may give more relevant evaluation, as some of the experimental measurements may have been made in non-physiological conditions or affected by significant error, associated with a poorly resolved structure, or indexed incorrectly in the database. They thus reported correlation and rmse of the predictions after excluding 10% of the predictions with most negative impacts on the correlation coefficient. Pires *et al.* (2014b) also reported their accuracy after 10% outlier removal. If we remove the 10% worst predicted stability effects from the combined predictions, we achieve correlation ρ of 0.92 and rmse of 0.67 kcal/mol. We report these results for all the methods in Supplementary Table S3 and also present the error distribution in Supplementary Figure S5.

The high accuracy is retained for variants with multiple mutations as well ($\rho = 0.88$ and rmse $= 1.33$ kcal/mol, see Table 2 and Supplementary Table S2). Table 3 lists mGPfusion's rmse for different number of simultaneous mutations. The model accuracy in fact improves up to 6 mutations. This is explained by the training set often containing the same single point mutations that appear in variants with multiple mutations. The model can then infer the combined effect of pointwise mutations. The model seems to fail when predicting the effects of 7–9 simultaneous mutations. Most of these mutations (8/12) are for Ribonuclease (1RGG) and their effects seem to be exceptionally difficult to predict. This may be because only few of the point mutations that are part of the multiple mutations are present in the training data. However, these mutations seem to be exceptionally difficult to predict for Rosetta as well, which could indicate that the experimental measurements concerning these mutations are not quite accurate. PoPMuSiC and mCSM are unable to predict multiple mutations, while Rosetta supports them, but its rmse accuracy decreases already with two mutations.

With multiple mutations, the decrease in performance between the position and mutation level cross-validations becomes clearer than with single mutations. With the position level cross-validation the stability effects of multiple mutations are predicted multiple times, which partly explains this loss of accuracy. For example, the effects of mutants with nine different simultaneous mutations, which were the most difficult cases in the mutation level cross-validation, are predicted nine times. Surprisingly, mGPfusion trained with protein level cross-validation achieves higher correlation and smaller errors than Rosetta; mGPfusion utilising simulated ΔΔG values for only single mutations, can predict the stability effects of multiple mutations better than Rosetta.

Downloaded from https://academic.oup.com/bioinformatics/article-abstract/34/13/i274/5045756
by Helsinki University of Technology user
on 14 August 2018

Table 3. Root-mean-square errors for different number of simultaneous mutations for all 15 proteins, with models trained by leave-one-out cross-validation

Mutations	1	2	3	4	5	6	7	8	9	10
Occurences	1211	207	52	42	4	8	3	3	6	1
mGPfusion	1.07	1.06	0.80	0.51	0.40	1.01	3.02	5.89	5.16	0.25
mGPfusion, only B62	1.11	1.12	0.77	0.59	0.29	1.14	3.00	6.78	5.56	0.11
mGP	1.04	1.03	0.61	0.50	0.18	0.92	3.23	6.18	6.75	0.08
mGP, only B62	1.26	0.96	0.65	0.83	0.26	1.14	2.95	6.90	6.57	0.05
Rosetta scaled	1.35	2.10	1.92	2.94	2.29	2.32	2.93	6.75	7.28	2.69
Rosetta	1.63	2.27	2.11	3.78	2.93	2.21	2.92	5.80	7.45	3.42

Note: Rosetta is added for comparison.

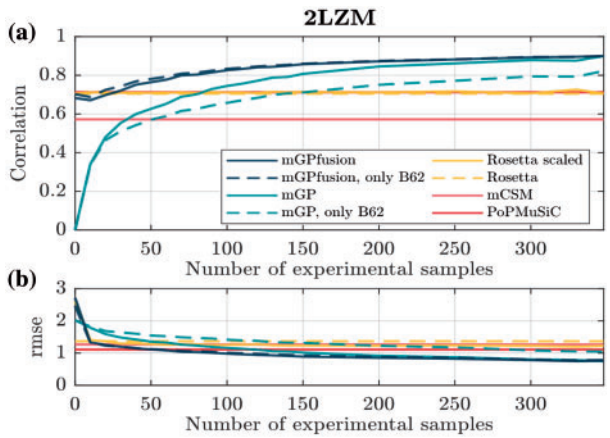


Fig. 4. (**a**) Correlation and (**b**) root mean square error of predictions made by models with different number of experimental training samples for T4 Lysozyme (2LZM). The results of Rosetta, mCSM and PoPMuSiC are invariant to training data (because mCSM and PoPMuSiC are pre-trained), and are thus constant lines. For both figures, an average of 100 randomly selected training sets is taken at each point

3.3 Uncertainty of the predictions

Gaussian processes provide a mean $\mu(\mathbf{x})$ and a standard deviation $\sigma(\mathbf{x})$ for the stability prediction of a protein variant $\mathbf{x}$. The standard deviation allows estimation of the prediction accuracy even without test data. Figure 1h visualizes the uncertainty of a few predictions made for the protein G (1PGA) when mutation level cross-validation is used. The estimated standard deviation allows a user to automatically identify low quality predictions that can appear e.g. in parts of the input protein space from which less data is included in model training. Conversely, in order to minimize the amount of uncertainty in the mGPfusion predictions, estimated standard deviation can be used to guide next experiments. The probabilistic nature of the predictions also admits an alternative error measure of negative log probability density (NLPD) $\text{nlpd} = -\sum_{i=1}^{N} \log p(y_i|\mu(\mathbf{x}_i), \sigma^2(\mathbf{x}_i))$, which can naturally take into account the prediction variance.

3.4 Effect of training set size

The results presented in Sections 3.1–3.3 used all available data for training with cross-validation to obtain unbiased performance measures. The inclusion of thousands of simulated variants allows the model to learn accurate models with less experimentally measured variants. Hence, we study how the mGPfusion model with or without simulated data performs with reduced number of experimental observations. To facilitate this, we randomly selected subsets of experimental data of size 0, 10, 20 and so on. We learned the mGP and mGPfusion models with these reduced experimental datasets while always using the full simulated datasets. This also allows us to estimate how the models work with different number of cross-validation folds. For example, the point of a learning curve which utilizes 2/3 or 4/5 of the training data correspond to an average of multiple 3-fold or 5-fold cross-validations, respectively.

The learning curve in Figure 4a shows how the averaged correlation for protein 2LZM improves when the size of the experimental dataset increases. The right-most values at $N = 348$ are obtained with leave-one-out cross-validation. The inclusion of simulated data in mGPfusion (dark blue line) consistently improves the performance of mGP, which is trained without simulated data. Figure 4b illustrate the difference in root mean square error. Learning curves for all proteins listed in Table 1 can be found from the Supplementary Figures S6–S8. When the number of experimental samples is zero, the mGPfusion model is trained solely using the simulated data with scaling $0.57y^S$, and the mGP model predicts the stability effect of every mutation as zero. The last point on the learning curves is obtained with mutation level cross-validation (see Table 2 and Supplementary Table S2).

3.5 Effect of data fusion and multiple substitution matrices

In the beginning of the learning curves, when only little training data is available, mGPfusion quite consistently outperforms the mGP model, demonstrating that the additional simulated data improves the prediction accuracy. However, when more training data becomes available, the performance of mGP model is almost as good or sometimes even better than the performance of the mGPfusion model. This shows that if enough training data is available, it is not necessary to simulate additional data in order to obtain accurate predictions. Table 2 also shows, that the data fusion can compensate the lack of relevant training data—with the mGPfusion models that utilize the additional data, the decrease in accuracy is smaller when position level cross-validation is used instead of mutation level cross-validation, than with the mGP models.

The varying weights for the base kernels between different proteins (shown in Fig. 2) already illustrated that different proteins benefit from different similarity measures for amino acid substitutions. The learning curves also support this observation—with some of the proteins mGPfusion model trained with only one kernel that utilizes BLOSUM62, provides approximately as good results as the mGPfusion model trained with multiple kernels. However, with many of the proteins, utilising just BLOSUM62 does not seem to be sufficient and the accuracy of the model can be improved by using different substitution matrices. Prior knowledge of appropriate

Downloaded from https://academic.oup.com/bioinformatics/article-abstract/34/13/i274/5045756
by Helsinki University of Technology user
on 14 August 2018

substitution models for each protein could enable creation of accurate prediction models with just one substitution model, but the MKL seems to be a good tool for selecting suitable substitution models when such knowledge is not available. It seems that the data fusion and number or relevance of used substitution matrices can compensate each other—the learning curves show, that the difference between mGPfusion models trained with one or multiple kernels is smaller than the difference between the mGP models utilising one or multiple kernels. This indicates that if additional simulated data is exploited, the use of multiple or appropriate substitution models is not as important than without the data fusion. On the other hand, if data fusion is not applied, the use of MKL can more significantly improve the accuracy of the mGP model.

3.6 Effect of the Bayesian transformation on Rosetta

The Bayesian scaling of simulated Rosetta values, proposed in Section 2.2, improves the match of Rosetta simulated values to empirical ΔΔG values even without using the Gaussian process framework. The Bayesian scaling improves the performance of standard Rosetta simulations from $\rho = 0.55$ and rmse = 1.63 kcal/mol to $\rho = 0.65$ and rmse = 1.35 kcal/mol (see Table 2 and Supplementary Table S2). This shows that the scaling proposed by Kellogg *et al.* (2011) indeed is not always the optimal scaling and significant improvements can be gained by optimising the scaling using a set of training data.

Figure 1g visualizes the Bayesian scaling for protein 1PGA, where the very destabilising ΔΔG values are dampened by the scaling (black dots) to less extreme values by matching the scaled simulated values to the experimental points (blue circles). The black dots along the scaling curve indicate the grid of point mutations after transformation. The scaling variance $\boldsymbol{\sigma}_T^2$ is indicated by the green region's vertical width, and on the right panel. The scaling tends to dampen very small values into less extreme stabilities, while it also estimates higher uncertainties for stability values further away from ΔΔG = 0. However, the scalings vary between different proteins, as can be seen from the transformations for each of the 15 proteins presented in Supplementary Figure S9.

4 Conclusions

We present a novel method mGPfusion for predicting stability effects of both single and multiple simultaneous mutations. mGPfusion utilizes structural information in form of contact maps and integrates that with amino acid residues and combines both experimental and comprehensive simulated measurements of mutations' stability effects. In contrast to earlier general-purpose stability models, mGPfusion model is protein-specific by design, which improves the accuracy but necessitates having a set of experimental measurements from the protein. In practise small datasets of 10–20 experimental observations were found to provide state-of-the-art accuracy models when supplemented by large simulation datasets.

An important advantage over most state-of-the-art machine learning methods is that mGPfusion is able to predict the effects of multiple simultaneous mutations in addition to single point mutations. Our experiments show that mGPfusion is reliable in predicting up to six simultaneous mutations in our dataset. Furthermore, the Gaussian process framework provide a way to estimate the (un)certainty of the predictions even without a separate test set. We additionally proposed a novel Bayesian scaling method to re-calibrate simulated protein stability values against experimental observations. This is a crucial part of the mGPfusion model, and also alone improved protein-specific Rosetta stability predictions by calibrating them using experimental data.

mGPfusion is best suited for a situation, where a protein is thoroughly experimented on and accurate predictions for stability effects upon mutations are needed. It takes some time to set up the framework and train the model, but after that new predictions can be made in fractions of a second. The most time-consuming part is running the simulations with Rosetta, at least when the most accurate protocol 16 is used. Simulating all 19 possible point mutations for one position took about 12 h, but simulations for different positions can be run on parallel. The time needed for training the prediction model depends on the amount of experimental and simulated training data. With no simulated data, the training time ranged from few seconds to few minutes. With data fusion and a single kernel, the training time was under an hour. With data fusion and MKL with 21 kernels, the training time was from a few minutes to a day.

Acknowledgements

We acknowledge the computational resources provided by the Aalto Science-IT.

Funding

This work was supported by the Academy of Finland Center of Excellence in Systems Immunology and Physiology, the Academy of Finland grants no. 260403 and 299915, and the Finnish Funding Agency for Innovation Tekes (grant no 40128/14, Living Factories).

Conflict of Interest: none declared.

References

Alberts,B. *et al.* (2007) *Molecular Biology of the Cell*, 5th edn. Garland Science, New York, USA.

Anslyn,E.V. and Dougherty,D.A. (2006) *Modern Physical Organic Chemistry*. University Science Books.

Berman,H.M. *et al.* (2000) The protein data bank. *Nucleic Acids Res.*, **28**, 235–242.

Bommarius,A.S. *et al.* (2011) Status of protein engineering for biocatalysts: how to design an industrially useful biocatalyst. *Curr. Opin. Chem. Biol.*, **15**, 194–200.

Branden,C. and Tooze,J. (1999) *Introduction to Protein Structure*, 2nd edn. Garland Science, New York, USA.

Capriotti,E. *et al.* (2005a) I-Mutant2.0: predicting stability changes upon mutation from the protein sequence or structure. *Nucleic Acids Res.*, **33**, W306–W310.

Capriotti,E. *et al.* (2005b) Predicting protein stability changes from sequences using support vector machines. *Bioinformatics*, **21**, ii54–ii58.

Capriotti,E. *et al.* (2008) A three-state prediction of single point mutations on protein stability changes. *BMC Bioinformatics*, **9**, S6.

Chen,C.-W. *et al.* (2013) iStable: off-the-shelf predictor integration for predicting protein stability changes. *BMC Bioinformatics*, **14**, S5.

Cheng,J. *et al.* (2005) Prediction of protein stability changes for single-site mutations using support vector machines. *Proteins Struct. Funct. Bioinf.*, **62**, 1125–1132.

Cherry,J.R. and Fidantsef,A.L. (2003) Directed evolution of industrial enzymes: an update. *Curr. Opin. Biotechnol.*, **14**, 438–443.

Cichonska,A. *et al.* (2017) Computational-experimental approach to drug-target interaction mapping: a case study on kinase inhibitors. *PLoS Comput. Biol.*, **13**, e1005678.

Dehouck,Y. *et al.* (2009) Fast and accurate predictions of protein stability changes upon mutations using statistical potentials and neural networks: poPMuSiC-2.0. *Bioinformatics*, **25**, 2537–2543.

Folkman,L. *et al.* (2014) Feature-based multiple models improve classification of mutation-induced stability changes. *BMC Genomics*, **15**, S6.

Downloaded from https://academic.oup.com/bioinformatics/article-abstract/34/13/i274/5045756
by Helsinki University of Technology user
on 14 August 2018

Giguere,S. *et al*. (2013) Learning a peptide-protein binding affinity predictor with kernel ridge regression. *BMC Bioinformatics*, **14**, 82.

Giollo,M. *et al*. (2014) NeEMO: a method using residue interaction networks to improve prediction of protein stability upon mutation. *BMC Genomics*, **15**, S7.

Henikoff,S. and Henikoff,J.G. (1992) Amino acid substitution matrices from protein blocks. *Proc. Natl. Acad. Sci. USA*, **89**, 10915–10919.

Kawashima,S. *et al*. (2007) AAindex: amino acid index database, progress report 2008. *Nucleic Acids Res*., **36**, D202–D205.

Kellogg,E.H. *et al*. (2011) Role of conformational sampling in computing mutation-induced changes in protein structure and stability. *Proteins Struct. Funct. Bioinf*., **79**, 830–838.

Kirk,O. *et al*. (2002) Industrial enzyme applications. *Curr. Opin. Biotechnol*., **13**, 345–351.

Kumar,M.S. *et al*. (2006) ProTherm and ProNIT: thermodynamic databases for proteins and protein–nucleic acid interactions. *Nucleic Acids Res*., **34**, D204–D206.

Leaver-Fay,A. *et al*. (2011) ROSETTA3: an object-oriented software suite for the simulation and design of macromolecules. *Methods Enzymol*., **487**, 545.

Liu,J. and Kang,X. (2012) Grading amino acid properties increased accuracies of single point mutation on protein stability prediction. *BMC Bioinformatics*, **13**, 44.

Menchetti,S. *et al*. (2005) Weighted decomposition kernels. In: *Proceedings of the 22nd International Conference on Machine Learning*. ACM, pp. 585–592.

Pace,C.N. and Scholtz,J.M. (1997) Measuring the conformational stability of a protein. *Protein Struct. Pract. Approach*, **2**, 299–321.

Pace,C.N. and Shaw,K.L. (2000) Linear extrapolation method of analyzing solvent denaturation curves. *Proteins Struct. Funct. Bioinf*., **41**, 1–7.

Pires,D.E. *et al*. (2014a) DUET: a server for predicting effects of mutations on protein stability using an integrated computational approach. *Nucleic Acids Res*., **42**, W314–W319.

Pires,D.E. *et al*. (2014b) mCSM: predicting the effects of mutations in proteins using graph-based signatures. *Bioinformatics*, **30**, 335–342.

Potapov,V. *et al*. (2009) Assessing computational methods for predicting protein stability upon mutation: good on average but not in the details. *Protein Eng. Des. Select*., **22**, 553–560.

Rapley,R. and Walker,J.M. (2000) *Molecular Biology and Biotechnology*, 4th edn. Royal Society of Chemistry, Cambridge, UK.

Rasmussen,C.E. and Williams,C.K.I. (2006) *Gaussian Processes for Machine Learning*. The MIT Press, Cambridge, MA.

Sanchez,S. and Demain,A.L. (2011) Enzymes and bioconversions of industrial, pharmaceutical, and biotechnological significance. *Organic Process Res. Dev*., **15**, 224–230.

Schmidt,M.W. *et al*. (2009). Optimizing costly functions with simple constraints: a limited-memory projected quasi-newton algorithm. In: *International Conference on Artificial Intelligence and Statistics*.

Shawe-Taylor,J. and Cristianini,N. (2004) *Kernel Methods for Pattern Analysis*. Cambridge University Press.

Tian,J. *et al*. (2010) Predicting changes in protein thermostability brought about by single- or multi-site mutations. *BMC Bioinformatics*, **11**, 370.

Tokuriki,N. and Tawfik,D.S. (2009) Stability effects of mutations and protein evolvability. *Curr. Opin. Struct. Biol*., **19**, 596–604.

Tomii,K. and Kanehisa,M. (1996) Analysis of amino acid indices and mutation matrices for sequence comparison and structure prediction of proteins. *Protein Eng. Des. Select*., **9**, 27–36.

Vishwanathan,S.V.N. *et al*. (2010) Graph kernels. *J. Mach. Learn. Res*., **11**, 1201–1242.

Wainreb,G. *et al*. (2011) Protein stability: a single recorded mutation aids in predicting the effects of other mutations in the same amino acid site. *Bioinformatics*, **27**, 3286–3292.

Downloaded from https://academic.oup.com/bioinformatics/article-abstract/34/13/i274/5045756
by Helsinki University of Technology user
on 14 August 2018