Computer Assisted Mechanics and Engineering Sciences, 13: 655–668, 2006.

Bayesian regression approaches on example of concrete fatigue failure prediction

Marek Słoński
Cracow University of Technology, Institute for Computational Civil Engineering
ul. Warszawska 24, 31-155 Kraków, Poland

(Received in the final form October 31, 2006)

The focus of this paper is the application of two nonlinear regression models in the context of Bayesian inference to the problem of failure prediction of concrete specimen under repeated loads based on experimental data. These two models are compared with an empirical formulae. Results on testing data show that both models give better point predictions than empirical formulae. Moreover, Bayesian regression approach makes it possible to calculate prediction intervals (error bars) describing the reliability of the models predictions.

Keywords: Bayesian inference, concrete, fatigue failure, Gaussian process regression, feed-forward neural network

1. Introduction

Fatigue of concrete is a process of progressive and permanent damage in concrete subjected to repeated loading. Evolution of this process is influenced by several factors, for example the concrete composition, the mechanical properties and the loading conditions [7].

In many cases, a prediction problem can be posed as a regression problem. The task is to infer a relationship between the input vector of variables and the output variable, based on a set of examples (experimental data). Feed-forward neural networks (FFNNs) and Gaussian process regression (GPR) are both nonlinear, very flexible regression models which are commonly used to approximate the unknown relationship in data.

GPR and FFNN models with Bayesian inference were used in similar problems of materials design and optimisation. Bailer-Jones *et al.* [1] used these two models for the prediction of the deformed and annealed microstructures of an aluminium alloy. Lampinen and Vehtari [9] used FFNNs and GPR for the prediction of the concrete quality properties.

Various feed-forward neural network models with classical, least-squares learning were used to the problem of predicting fatigue failure [15]. For example, Waszczyszyn *et al.* used a back propagation neural network (BPNN) [8]. Jakubek applied a fuzzy weights neural network (FWNN) [6]. This paper is an extension of the earlier work published in [13] where we compared 4-10-1 Bayesian FFNN model with Hybrid Monte Carlo approximation of Bayesian learning and prediction and an early-stopped committee of FFNN models trained based on maximum likelihood approach and scaled conjugate gradient optimization algorithm and using 10-fold cross validation for estimation of the predictive accuracy.

The problem addressed in this paper is the prediction of the concrete fatigue failure based on GPR model and FFNN models with simple, deterministic approximations of Bayesian learning and prediction. These models are defined and fitted to the experimental data using Bayesian inference and then applied to the prediction of fatigue failure. The paper has the following structure: in Sec. 2, classical and Bayesian approaches to learning and prediction are presented, in Sec. 3. two Bayesian regression models are described, in Sec. 4, the problem of the concrete fatigue failure prediction is

defined and in Sec. 5, the numerical experiments and results are presented. In the last section the final conclusions are stated.

2. Bayesian regression

In this section, we introduce parametric and nonparametric models for regression. We also describe two approaches to learning and predictions following the description of Bayesian inference for parametric models given by Bishop and Tipping in [4, 14]. In the parametric approach to regression, we represent the underlying relationship with some function $y(\mathbf{x};\mathbf{w})$ which is parameterised by a finite number of parameters $\mathbf{w}$. A feed-forward neural network model is an example of a parametric regression model. A Gaussian process model is a nonparametric model. The nonparametric regression models $y(\mathbf{x})$ are defined without an explicit parameterisation. They are defined using the set of hyperparameters. One well known nonparametric regression method is the spline smoothing method.

2.1. Classical approach to learning

For regression problems it is generally assumed that a target variable t is the sum of an underlying deterministic function $y(\mathbf{x})$ and a random variable ϵ,

$$t_n = y(\mathbf{x}_n;\mathbf{w}) + \epsilon_n\,, \tag{1}$$

where t_n is the target value for the corresponding input vector $\mathbf{x}_n$, $y(\mathbf{x}_n;\mathbf{w})$ is the model output value and ϵ_n is the noise component. Based on the training data set $\mathcal{D} = \{(\mathbf{x}_1,t_1),(\mathbf{x}_2,t_2),\ldots,(\mathbf{x}_n,t_n,),\ldots,(\mathbf{x}_N,t_N,)\}$ the model is trained, i.e. the unknown parameters are estimated by minimising some error (loss) function, for example a sum of squared errors function

$$E_D(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}[t_n - y(\mathbf{x}_n;\mathbf{w})]^2\,, \tag{2}$$

giving a point estimate of the unknown parameter vector called a 'least squares' estimate (LS), $\mathbf{w}_{LS}$. The well-known problem with this error function minimisation is that the complex models can 'over-fit' the training data giving unsatisfactory prediction on the testing data. One common approach to this problem is adding a regularization (penalty) term to the error function which controls the complexity (smoothness) of the trained model

$$E(\mathbf{w}) = E_D(\mathbf{w}) + \lambda E_W(\mathbf{w}), \tag{3}$$

where the regularization term is commonly assumed as the sum of squared-weight penalty term,

$$E_W(\mathbf{w}) = \frac{1}{2}\sum_{m=1}^{M} w_m^2\,, \tag{4}$$

where M is the number of parameters and λ is a hyperparameter which controls the trade-off between the fit of the model to the training data and the smoothness of the estimated relationship. In non-Bayesian approach, this hyperparameter is set to the value for which the error calculated on the validation data set (i.e. data not used to estimate $\mathbf{w}$) is minimal. Finally, we obtain a point estimate which is known as a 'penalised least squares' estimate (PLSE), $\mathbf{w}_{PLS}$.

2.2. Bayesian learning

In Bayesian approach to regression, we start with definition of a probabilistic model of noise term ϵ_n and a prior distribution over unknown parameters of the model $\mathbf{w}$. Here we assumed the additive noise model to be a Gaussian distribution defined by two parameters: mean μ and variance σ^2, i.e. $p(\epsilon|\mu,\sigma^2) = N(\epsilon|\mu,\sigma^2)$. Moreover, we assume that the noise has zero mean and is independent and identically distributed. Finally, the Gaussian noise model can be written as [4]

$$N(\epsilon|\mu,\sigma^2) = N(\epsilon|0,\sigma^2) = (2\pi\sigma^2)^{-1/2}\exp\left[-\frac{\epsilon^2}{2\sigma^2}\right]. \tag{5}$$

Based on this noise model and the assumed relation between the target value and the model output value represented in (1), the conditional distribution of the target variable given the input variable and the parameters $\mathbf{w}$ is also a Gaussian distribution

$$p(t|\mathbf{x},\mathbf{w},\sigma^2) = (2\pi\sigma^2)^{-1/2}\exp\left[-\frac{\{t-y(x;\mathbf{w})\}^2}{2\sigma^2}\right]. \tag{6}$$

The joint probability of the data set $\mathbf{t}$ is given by the product over all data points of the distribution (6) evaluated at the observed data points (since the data points are independent)

$$p(\mathbf{t}|\mathbf{x},\mathbf{w},\sigma^2) = \prod_{n=1}^{N} p(t_n|x_n,\mathbf{w},\sigma^2) = \prod_{n=1}^{N}(2\pi\sigma^2)^{-1/2}\exp\left[-\frac{\{t_n-y(x_n;\mathbf{w})\}^2}{2\sigma^2}\right]. \tag{7}$$

This joint probability distribution is called the likelihood function (when treated as a function of the parameters vector $\mathbf{w}$). The inverse of variance parameter is called precision and is defined by $\beta = 1/\sigma^2$.

The prior distribution is also assumed to be a spherical Gaussian distribution with zero mean and inverse variance (precision) hyperparameter α, given by

$$p(\mathbf{w}|\alpha) = \prod_{m=1}^{M}\left(\frac{\alpha}{2\pi}\right)^{1/2}\exp\left\{-\frac{\alpha}{2}w_m^2\right\}. \tag{8}$$

This prior distribution describes our a priori preferences for smoother models (such models should have smaller parameters values). Parameters such as α and β are often called hyperparameters since they control parameters of probability distributions over model parameters.

After defining the likelihood (7) and the prior distribution (8), we apply the Bayes' theorem and compute the posterior distribution over parameters

$$\texttt{posterior} = \frac{\texttt{likelihood} \times \texttt{prior}}{\texttt{normalising factor}}, \qquad p(\mathbf{w}|\mathbf{t},\alpha,\beta) = \frac{p(\mathbf{t}|\mathbf{w},\beta)p(\mathbf{w}|\alpha)}{p(\mathbf{t}|\alpha,\beta)}. \tag{9}$$

The normalising factor in (9) is called evidence. It ensures that the posterior distribution integrates to one and is given by an integral over the parameter space

$$p(\mathbf{t}|\alpha,\beta) = \int p(\mathbf{t}|\mathbf{w},\beta)\,p(\mathbf{w}|\alpha)\,\mathrm{d}\mathbf{w}. \tag{10}$$

The Bayes' rule allowed us to combine two sources of information about the estimated relationship and instead of the point estimate, we have obtained the updated distribution (in the light of the training data) over all possible values of parameters.

Finally, we can notice that the two non-Bayesian approaches to parameters estimation described above, (2), (3), can be regarded as the approximations to the full Bayesian approach, (9). The

standard, 'least-square' point estimate (LS) for $\mathbf{w}_{LS}$ can be obtained equivalently by minimising the negative logarithm of the likelihood function, (7) with respect to weights $\mathbf{w}$, given by

$$E_{ML}(\mathbf{w}) = -\log p(\mathbf{t}|\mathbf{x}, \mathbf{w}, \beta) = \frac{N}{2}\log(2\pi\beta) + \frac{\beta}{2}\sum_{n=1}^{N}\{t_n - y(x_n; \mathbf{w})\}^2. \tag{11}$$

This point estimate is called in classical statistics a 'maximum likelihood' estimate (ML) and is denoted by $\mathbf{w}_{ML}$. We can also minimise (7) with respect to β and obtain the following estimate of the noise level

$$\frac{1}{\beta_{ML}} = \frac{1}{N}\sum_{n=1}^{N}|t_n - y(\mathbf{x}_n; \mathbf{w}_{ML})|^2. \tag{12}$$

Similarly, the regularised 'least-square' estimate (PLS) for the parameters $\mathbf{w}_{PLS}$ can be derived by minimising the negative logarithm of the numerator in the Bayes' rule, (9) given by

$$E_{MAP}(\mathbf{w}) = -\log p(\mathbf{t}|\mathbf{w}, \beta) - \log p(\mathbf{w}|\alpha) = \frac{\beta}{2}\sum_{n=1}^{N}\{t_n - y(x_n; \mathbf{w})\}^2 + \frac{\alpha}{2}\sum_{m=1}^{M} w_m^2, \tag{13}$$

where λ in Eq. (3) is given by $\lambda = \alpha/\beta$. This point estimate is known as a 'maximum a posteriori' (MAP) estimate and is denoted by $\mathbf{w}_{MAP}$.

2.3. Making predictions

After learning, we can make a prediction for the value of t_* given the new input vector $\mathbf{x}_*$. The key distinction between classical approach to prediction and Bayesian one is using marginalisation. In Bayesian approach, instead of using the single estimate of model parameters, we integrate them out. In the non-Bayesian approach with regularization, the point prediction is given by $y(\mathbf{x}_*; \mathbf{w}_{PLS})$. In the MAP Bayesian settings, instead of a point prediction, we have a predictive distribution $p(t_*|\mathbf{w}_{MAP}, \beta)$, taking into account the MAP point estimate for weights $\mathbf{w}_{MAP}$. Finally, in the fully Bayesian approach, we obtain a predictive distribution by integrating out parameters $\mathbf{w}$, i.e. averaging the model probability for t_* over all possible values of $\mathbf{w}$. The predictive distribution given a new input vector is given by

$$p(t_*|\mathbf{t}, \alpha, \beta) = \int p(t_*|\mathbf{w}, \beta)\, p(\mathbf{w}|\mathbf{t}, \alpha, \beta)\, \mathrm{d}\mathbf{w}. \tag{14}$$

This distribution incorporates uncertainty over parameters in the light of training data.

So far we have treated the hyperparameters α and β as we have known them. But in the fully Bayesian framework, we should take into account also uncertainty over the hyperparameters integrate them out. We first define the prior distributions over the hyperparameters $p(\alpha)$ and $p(\beta)$, which are called hyperprior. Having defined these priors we obtain the full posterior distribution over weights

$$p(\mathbf{w}, \alpha, \beta|\mathbf{t}) = \frac{p(\mathbf{t}|\mathbf{w}, \beta)\, p(\mathbf{w}|\alpha)\, p(\alpha)\, p(\beta)}{p(\mathbf{t})}, \tag{15}$$

where the denominator (normalising term) in (15) is

$$p(\mathbf{t}) = \int p(\mathbf{t}|\mathbf{w}, \beta)\, p(\mathbf{w}|\alpha)\, p(\alpha)\, p(\beta)\, \mathrm{d}\mathbf{w}\, \mathrm{d}\alpha\, \mathrm{d}\beta. \tag{16}$$

Finally, we can make a fully Bayesian prediction based on the posterior distribution (15), taking into account all possible values of model parameters and the hyperparameters

$$p(t_*|\mathbf{t}) = \int p(t_*|\mathbf{w}, \beta)\, p(\mathbf{w}, \alpha, \beta|\mathbf{t})\, \mathrm{d}\mathbf{w}\, \mathrm{d}\alpha\, \mathrm{d}\beta. \tag{17}$$

Nevertheless, this integral can not be computed analytically and we must approximate the integrations using one of the methods developed so far [14]. These methods can be divided into two groups: deterministic (type 2 maximum likelihood, Laplace's method, variational techniques) and stochastic (Monte Carlo methods). The detailed description of these methods can be found in Bishop's book [3] or MacKay's book [10].

2.4. The evidence approximation

In this paper we use the deterministic approach based on type 2 maximum likelihood which is called also evidence approximation (EA) for α and β [2]. This approach starts with rewriting the full posterior distribution (15), using the product rule of probability as:

$$p(\mathbf{w}, \alpha, \beta|\mathbf{t}) = p(\mathbf{w}|\mathbf{t}, \alpha, \beta)\, p(\alpha, \beta|\mathbf{t}). \tag{18}$$

The first term is the posterior distribution defined in Eq. (9). The second term is the posterior distribution for hyperparameters

$$p(\alpha, \beta|\mathbf{t}) = \frac{p(\mathbf{t}|\alpha, \beta)\, p(\alpha)\, p(\beta)}{p(\mathbf{t})}. \tag{19}$$

This posterior distribution can be approximated by a δ-function at its mode. This approximation is valid if we assume that this distribution is sharply peaked around their most probable values α_{MP} and β_{MP}. Then the predictive distribution in Eq. (17) can be given by

$$p(t_*|\mathbf{t}) \simeq p(t_*|\mathbf{t}, \alpha_{MP}, \beta_{MP}) = \int p(t_*|\mathbf{w}, \beta_{MP})\, p(\mathbf{w}|\mathbf{t}, \alpha_{MP}, \beta_{MP})\, \mathrm{d}\mathbf{w}. \tag{20}$$

In order to find these values, we need to compute values of the hyperparameters which maximise the posterior (19). If we have no idea of suitable values for α and β then we can use relatively flat prior distributions $p(\alpha)$ and $p(\beta)$. In this case, the most probable values of the hyperparameters are obtained by maximising $p(\mathbf{t}|\alpha, \beta)$ term which in Eq. (9) is called evidence (also known as marginal likelihood function) and is given in Eq. (10) by

$$p(\mathbf{t}|\alpha, \beta) = \int p(\mathbf{t}|\mathbf{w}, \beta)\, p(\mathbf{w}|\alpha)\, \mathrm{d}\mathbf{w}. \tag{21}$$

The maximisation of the evidence function can only be done using another approximation. An approach called evidence approximation, uses the posterior distribution over weights approximated by a spherical Gaussian distribution around a mode of the posterior distribution. Then we can find values of the hyperparameters which maximise the evidence by differentiating the log of the evidence given by

$$\ln p(\mathbf{t}|\alpha, \beta) \simeq -E(\mathbf{w}_{MAP}) - \frac{1}{2}\ln|\boldsymbol{A}| + \frac{M}{2}\ln\alpha + \frac{N}{2}\ln\beta - \frac{N}{2}\ln(2\pi) \tag{22}$$

with respect to α and β and setting the derivatives to zero. The matrix $\boldsymbol{A}$ is the matrix of second derivatives of the negative log posterior distribution and is given by

$$\boldsymbol{A} = -\nabla\nabla \ln p(\mathbf{w}|\mathbf{t}, \alpha, \beta) = \alpha\boldsymbol{I} + \beta\boldsymbol{H} \tag{23}$$

where $\boldsymbol{H}$ is the Hessian matrix with values of the second derivatives of the error function (2), evaluated at $\mathbf{w} = \mathbf{w}_{MP}$. This procedure leads to the following formulae for finding the optimal values of the hyperparameters given by

$$\alpha^{\text{new}} = \frac{\gamma}{2E_W}, \tag{24}$$

$$\beta^{\text{new}} = \frac{N - \gamma}{2E_D}, \tag{25}$$

where γ is the number of well-determined parameters given by

$$\gamma = \sum_{i=1}^{M} \frac{\lambda_i}{\lambda_i + \alpha} \tag{26}$$

where λ_i is the eigenvalue of the Hessian $\boldsymbol{H}$ matrix, evaluated at $\mathbf{w} = \mathbf{w}_{MP}$. Similarly, the error terms E_W and E_D are evaluated, setting weights to the most probable values $\mathbf{w}_{MP}$. The details of the evidence approximation approach can be found in Bishop's book [3].

3. Nonlinear models for regression

In this section we briefly describe two nonlinear regression models: feed-forward neural network known as a multilayer perceptron (MLP) and Gaussian process regression (GPR). More details on these models can be found in two recently published textbooks by Bishop [3] and by Rasmussen and Williams [12].

We start by considering a broad class of functions called linear regression models. These models are linear functions in parameters $\mathbf{w}$ and nonlinear functions of the input vector $\mathbf{x}$. In general, linear models are defined as linear combinations of fixed, nonlinear basis functions of the input variables, of the form

$$y(\mathbf{x}; \mathbf{w}) = \sum_{m=1}^{M-1} w_m \phi_m(\mathbf{x}) + w_0 \tag{27}$$

where $\phi_m(\mathbf{x})$ are called basis functions. Equation (27) is often written by using an additional dummy 'basis function' $\phi_0(\mathbf{x}) = 1$

$$y(\mathbf{x}; \mathbf{w}) = \sum_{m=0}^{M-1} w_m \phi_m(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}). \tag{28}$$

There are many possible choices for the basis functions, for example, we can choose polynomial basis functions $\phi_m(x) = x^m$.

Linear regression models have some limitations as models for regression and we can overcome these limitations by allowing the basis functions to be adaptive. This approach leads, for example, to multilayer perceptron (MLP) neural network model.

Another class of models known as kernel methods can be obtained by reformulation of linear models in terms of dual representation. In this approach, linear regression models are trained by minimising a regularised error function (3), described in terms of the Gram matrix $\boldsymbol{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^T$. The Gram matrix is an $N \times N$ symmetric matrix with elements

$$K_{nm} = \boldsymbol{\phi}(\mathbf{x}_n)^T \boldsymbol{\phi}(\mathbf{x}_m) = k(\mathbf{x}_n, \mathbf{x}_m) \tag{29}$$

where $k(\mathbf{x}, \mathbf{x}') = \boldsymbol{\phi}(\mathbf{x})^T \boldsymbol{\phi}(\mathbf{x}')$ is a kernel function. The design matrix is an $N \times M$ matrix with elements $\boldsymbol{\Phi}_{nm} = \phi_m(x_n)$. The vector $\boldsymbol{k}(\mathbf{x})$ is defined with elements $k_n(\mathbf{x}) = k(\mathbf{x}_n, \mathbf{x})$.

The prediction for a new input $\mathbf{x}$ is obtained from

$$y(\mathbf{x}) = \boldsymbol{k}(\mathbf{x})^T (\boldsymbol{K} + \lambda \boldsymbol{I}_N)^{-1} \mathbf{t} \tag{30}$$

where $\mathbf{t} = (t_1, \ldots, t_N)^T$ is a vector of training target values, $\boldsymbol{K} = \boldsymbol{\Phi}\boldsymbol{\Phi}^T$ is the Gram matrix and $\boldsymbol{\Phi}$ is the design matrix. From the Bayesian viewpoint of linear regression models, dual representation approach leads to the Gaussian process regression (GPR) model, where the kernel function is interpreted as a covariance function of the Gaussian process.

3.1. Bayesian neural networks

A feed-forward neural network model can be treated as a parametric model parameterised by a vector of adaptive weights. The MLP neural network model with d inputs x_i, one hidden layer of H nonlinear units and a single, linear output can be expressed as a following equation

$$y(\mathbf{x};\mathbf{w}) = \sum_{j=1}^{H} w_j g\left(\sum_{i=1}^{d} w_{ji}x_i + w_{j0}\right) + w_0\,. \tag{31}$$

Nonlinear function $g()$ is an activation function of the hidden units and $\mathbf{w}$ is the parameters vector with: w_{ji} being the first layer weights from the ith input to the jth hidden unit and w_j being the second layer weights from jth hidden unit to the output. w_{j0} and w_0 are the bias parameters for the hidden and output unit respectively. In this paper we use 'tanh' activation function

$$g(a) = \frac{\exp(a) - \exp(a)}{\exp(a) + \exp(a)}\,. \tag{32}$$

In the Bayesian approach to neural networks learning, all weights are treated as random variables. As described in Sec. 2, we first define prior distribution $p(\mathbf{w}|\alpha)$ over weights which expresses our beliefs about the parameters before the data is observed. We also define noise model $p(\epsilon_n|\sigma^2)$, describing how the dependent variable may be corrupted by the noise, and finally we obtain the likelihood $p(\mathbf{t}|\mathbf{x},\mathbf{w},\sigma^2)$. Once we observe the data, Bayes' theorem is used to update our beliefs and we obtain the posterior distribution over weights $p(\mathbf{w}|\mathbf{t},\alpha,\sigma^2)$ which we use to make the prediction for a new input vector. The full description of the Bayesian techniques for neural networks can be found in Bishop's book [2].

3.2. Gaussian processes

In the introduction to this section we ended with a statement that the Gaussian process model can be derived from the Bayesian viewpoint of linear regression models. Here we concentrate on the key result allowing direct application of Gaussian process regression model for prediction. This is the predictive distribution of the target variable t_{N+1} for a new input vector $\mathbf{x}_{N+1}$. This requires evaluation of conditional distribution $p(t_{N+1}|\mathbf{t}_N)$, where $\mathbf{t}_N$ is a vector of training target values. This conditional distribution for the Gaussian processes is a Gaussian distribution with mean and covariance given by

$$m(\mathbf{x}_{N+1}) = \boldsymbol{k}^T \boldsymbol{C}_N^{-1}\mathbf{t}, \tag{33}$$

$$\sigma^2(\mathbf{x}_{N+1}) = c - \boldsymbol{k}^T \boldsymbol{C}_N^{-1}\boldsymbol{k}, \tag{34}$$

where the $\boldsymbol{C}_N$ is the $N \times N$ covariance matrix with elements given by a sum of two terms: the covariance function $k(\mathbf{x}_n,\mathbf{x}_m)$ and the Gaussian noise component represented by a precision β

$$C(\mathbf{x}_n,\mathbf{x}_m) = k(\mathbf{x}_n,\mathbf{x}_m) + \beta^{-1}\delta_{nm}\,. \tag{35}$$

The vector $\boldsymbol{k}$ has elements $k(\mathbf{x}_n,\mathbf{x}_{N+1})$ and the scalar c is $k(\mathbf{x}_{N+1},\mathbf{x}_{N+1}) + \beta^{-1}$. From Eqs. (33) and (34) we see that the Gaussian process regression model is completely defined by the covariance function $k(\mathbf{x}_n,\mathbf{x}_m)$. This function allows us to define the situation that for the nearby points $\mathbf{x}_n$ and $\mathbf{x}_m$ in the input space, the corresponding values $y(\mathbf{x}_n)$ and $y(\mathbf{x}_m)$ will be more strongly correlated than for dissimilar points [3].

The covariance function can be any function that will generate a non-negative definite covariance matrix for any ordered set of (input) vectors $(\mathbf{x}_1,\ldots,\mathbf{x}_n)$. It is usual to choose the covariance function to be stationary, i.e. such that the condition

$$k(\mathbf{x},\mathbf{x}') = k(\mathbf{x}-\mathbf{x}') \tag{36}$$

holds. This means that the location of the points $\mathbf{x}$ and $\mathbf{x}'$ does not affect their covariance, just the vector joining them. In this paper we use squared exponential (SE) covariance function

$$k(\mathbf{x}_n, \mathbf{x}_m) = \theta_0 \exp\left(-\frac{1}{2}\sum_{i=1}^{d} \eta_i (x_{ni} - x_{mi})^2\right) + \theta_2 \tag{37}$$

which is the exponential of a weighted squared distance between points in $\mathbb{R}^d$. The SE covariance function has some free parameters which are called hyperparameters to emphasise that they are parameters of a nonparametric model. The term θ_2 controls the vertical offset of the GPR model, while θ_0 controls the vertical scale of the process. The η_i hyperparameters allow a different distance measure for each dimension.

After defining the covariance function we can make predictions for the new input vectors but it is often necessary to learn the hyperparameters before making reliable prediction. The simplest approach is similar to the evidence approximation discussed above. For Gaussian process regression, we find the most probable hyperparameters of the covariance function by maximising the log likelihood function given by

$$\ln p(\mathbf{t}|\theta) = -\frac{1}{2}\ln|\boldsymbol{C}_N| - \frac{1}{2}\mathbf{t}^T\boldsymbol{C}_N^{-1}\mathbf{t} - \frac{N}{2}\ln(2\pi), \tag{38}$$

using gradient-based optimisation algorithms such as conjugate gradients. More details on Gaussian process regression models can be found in Bishop's book [3] and in the book by Rasmussen and Williams [12].

4. Concrete fatigue failure

Concrete fatigue failure can be defined as a number of loading cycles N causing fatigue damage of plain concrete specimens. The problem of predicting concrete fatigue failure was formulated as a Bayesian regression problem. We assumed that the fatigue failure is a sum of an underlying deterministic function $y(\mathbf{x})$ and a random variable ϵ. The input vector $\mathbf{x}$ consists of four explanatory variables, namely concrete static uniaxial compressive strength (f_c), ratio of minimal and maximal stress level in compressive cycle of loading ($R = \sigma_{\min}/\sigma_{\max}$), ratio of compressive fatigue and static strength of concrete, also called maximal compressive stress level ($\chi = f_{cN}/f_c$) and frequency of the loading cycle (f). The target variable is the scalar output $y = \log N$.

4.1. Data set

In Ref. [5] a wide experimental evidence was described and compiled, corresponding to more than 400 tests performed in 14 laboratories. The concrete specimens were subjected to cycles of compressive loadings and the numbers of cycles N which caused the specimens fatigue damage were measured. In this paper we used only $P = 218$ results (examples) of tests from 8 laboratories mentioned in [6] and [8]. For example, in Fig. 1 the results of two various fatigue tests on concrete samples are presented. In Table 1 the statistical parameters, namely minimal and maximal values, mean values and standard deviations for inputs and output variables are shown.

Table 1. Statistical parameters of input and output variables

Variable	Min	Max	Mean	St. Dev.
f_c [MPa]	20.70	45.20	34.68	8.84
R [–]	0.00	0.88	0.14	0.18
f [Hz]	0.025	150.0	21.30	39.38
χ [–]	0.49	0.94	0.74	0.11
$\log N$ [–]	1.86	7.34	4.56	1.41

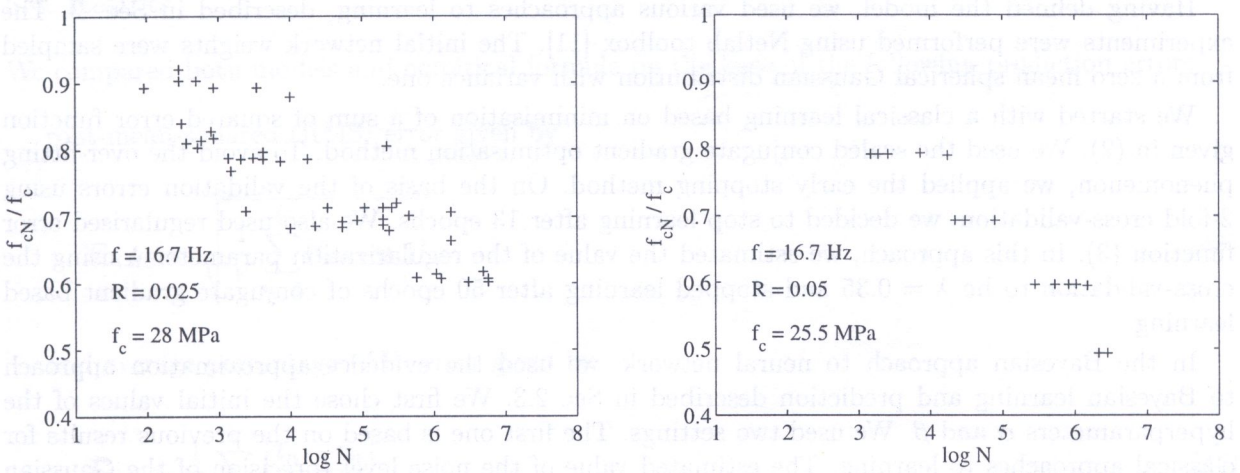

Fig. 1. Results of fatigue tests on concrete samples given by: (left) Artim and McLaughlin (1959), (right) Gray *et al.* (1961) taken from [5]

5. Numerical experiments and results

In this section we present numerical experiments using neural network and Gaussian process models which we applied to the fatigue tests data. The learning was performed using $L = 118$ learning examples (patterns). The generalisation capability of the models in predicting concrete fatigue failure was estimated using $T = 100$ testing examples.

In the analysis both the inputs and output variables were first standardised to zero mean and unit standard deviation by transformation

$$\tilde{x}_i^n = \frac{x_i^n - \bar{x}_i}{s_i}, \tag{39}$$

where $\bar{x}_i$ is an average value and s_i is the standard deviation

$$\bar{x}_i = \frac{1}{N}\sum_{n=1}^{N} x_i^n, \qquad s_i = \sqrt{\frac{1}{N-1}\sum_{n=1}^{N}(x_i^n - \bar{x}_i)^2}. \tag{40}$$

This data transformation can be viewed as a form of preconditioning technique which remove the possible source of ill-conditioning of learning.

5.1. Neural network

A feed-forward neural network with a single hidden layer of hyperbolic tangent units (neurons) and linear output unit was used to model the relationship between the inputs and the output variables. On the basis of the preliminary analysis, we decided to use 15 hidden units. This number of units should give the neural model sufficient flexibility in approximating the relationship. The total number of parameters of 4-15-1 neural network is 91 which is smaller than the number of learning patterns $L = 118$. This model can be expressed by the following equation

$$y(\mathbf{x};\mathbf{w}) = \sum_{j=1}^{15} w_j \tanh\left(\sum_{i=1}^{4} w_{ji}x_i + w_{j0}\right) + w_0. \tag{41}$$

Having defined the model, we used various approaches to learning, described in Sec. 2. The experiments were performed using Netlab toolbox [11]. The initial network weights were sampled from a zero mean spherical Gaussian distribution with variance one.

We started with a classical learning based on minimisation of a sum of squared error function given in (2). We used the scaled conjugate gradient optimisation method. To avoid the over-fitting phenomenon, we applied the early stopping method. On the basis of the validation errors using 2-fold cross-validation, we decided to stop learning after 13 epochs. We also used regularised error function (3). In this approach, we estimated the value of the regularization parameter λ using the cross-validation to be $\lambda = 0.35$ and stopped learning after 50 epochs of conjugate gradient based learning.

In the Bayesian approach to neural network, we used the evidence approximation approach to Bayesian learning and prediction described in Sec 2.3. We first chose the initial values of the hyperparameters α and β. We used two settings. The first one is based on the previous results for classical approaches to learning. The estimated value of the noise level (precision of the Gaussian noise model) was computed from Eq. (12), $\beta = 2.1217$. The value of the hyperparameter α was computed knowing that $\lambda = \alpha/\beta$, which gives $\alpha = 0.7426$. On the base of the cross-validation, we decided to stop the re-estimation procedure for hyperparameters given by Eqs. (24)–(26) after only one iteration. The final values of hyperparameters in the first case are $\beta = 2.8665$, $\alpha = 2.7554$ and $\gamma = 42.1395$ gives the number of well-determined parameters. In the second settings, we used the commonly assumed initial values for hyperparameters $\alpha = 0.01$ and $\beta = 100$. After one iteration of the re-estimation procedure, the final values of hyperparameters are $\beta = 2.2162$, $\alpha = 2.6358$ and $\gamma = 60.9431$

5.2. Gaussian process

A Gaussian process regression model with a squared exponential covariance function was applied to model the data set. The hyperparameters of the covariance function were estimated using maximum likelihood approach described in Sec. 3 and the conjugate gradient optimisation method with only 6 iteration (trying to avoid over-fitting as stated based on 2-fold cross-validation). The Netlab toolbox [11] implementation of GPR model was used to perform the experiments.

5.3. Empirical formula

In ref. [5] an empirical formula was derived by Furtak as the following implicit relation between variables:

$$\log N = \frac{1}{A}\Big[\log(1.16 \cdot C_f/\chi) + \log(1 + B \cdot R \cdot \log N)\Big] \tag{42}$$

where: $\chi = f_{cN}/f_c$ i $R = \sigma_{\min}/\sigma_{\max}$ and the parameters according to ref. [5] have the following values,

$$\begin{aligned} A &= 0.008 - 0.118 \cdot \log(\sigma_I/f_c), \\ B &= 0.118 \cdot (\sigma_{II}/\sigma_I - 1), \\ C_f &= 1 + 0.07 \cdot (1 - R) \cdot \log f, \end{aligned}$$

σ_I and σ_{II} are critical strengths.

5.4. Results

We compared both models and empirical formula on the base of the following prediction errors:

- root-mean-squared (RMS) error given by

$$E_{RMS} = \sqrt{\frac{1}{V}\sum_{n=1}^{V}(t_n - y_n)^2}\,, \tag{43}$$

- and average percentage (AP) error given by

$$E_{AP} = \frac{1}{V}\sum_{n=1}^{V}\left|\frac{t_n - y_n}{t_n}\right|. \tag{44}$$

Also the coefficient of correlation r was computed

$$r = \frac{\sum_{n=1}^{V}(t_n - \bar{t})(y_n - \bar{y})}{\sum_{n=1}^{V}(t_n - \bar{t})\,\sum_{n=1}^{V}(y_n - \bar{y})}\,, \tag{45}$$

where $\bar{t}$ and $\bar{y}$ are mean values of targets t_n and predicted values y_n, respectively.

The computed RMS errors for all $P = 218$ patterns as well as for $L = 118$ training and $T = 100$ testing patterns for various variants of MLP neural network training and Gaussian process regression model are presented in Table 2. There are also shown average percentage (AP) errors as well as coefficients of correlation r. The results for testing data indicate that the Gaussian process regression (GPR) model has slightly better generalization properties ($E_{RMS}(T) = 0.771$, $r(T) = 0.856$) than the MLP model with various methods of the parameters estimation.

Table 2. Comparison of generalisation performance of multilayer perceptron (MLP) neural network models with various Bayesian learning approaches (maximum likelihood – ML, maximum a posteriori – MAP, evidence approximation – EA with two different sets (s1, s2) of initial values of the hyperparameters) and Gaussian process regression (GPR) model with maximum likelihood – ML approach to estimation of the covariance function parameters in predicting concrete fatigue failure. In third column the estimated standard deviation of the Gaussian noise model is presented

Model	Learning	Noise	RMS(V)			AP(V)		r(V)		
		σ	P	L	T	L	T	P	L	T
Formula	–	–	0.931	0.873	0.991	17.7%	26.3%	0.843	0.843	0.843
MLP	ML	0.69	0.732	0.686	0.782	12.8%	20.8%	0.864	0.876	0.849
MLP	MAP	0.67	0.724	0.670	0.784	12.6%	20.6%	0.868	0.883	0.852
MLP	EA, s1	0.59	0.723	0.675	0.775	12.7%	20.4%	0.868	0.881	0.853
MLP	EA, s2	0.67	0.718	0.666	0.776	12.6%	20.6%	0.869	0.884	0.852
GPR	ML	0.51	0.724	0.681	0.771	12.9%	20.4%	0.868	0.879	0.856

The predicted influence of maximal stress level on the fatigue failure of concrete by MLP models and GPR model are presented in Figs. 2 and 3, as well as error bars. There are also shown the actual measured values, for two sets of tests, namely tests performed by Artim and McLaughlin and tests performed by Gray et al. Both models give similar predictions for the mean value of the predictive Gaussian distribution. However, there is a significant discrepancy between MLP and GPR predicted error bars (confidence intervals). Gaussian process model has much smaller variance of the predictive distribution than MLP networks. In this context, it is interesting to compare the

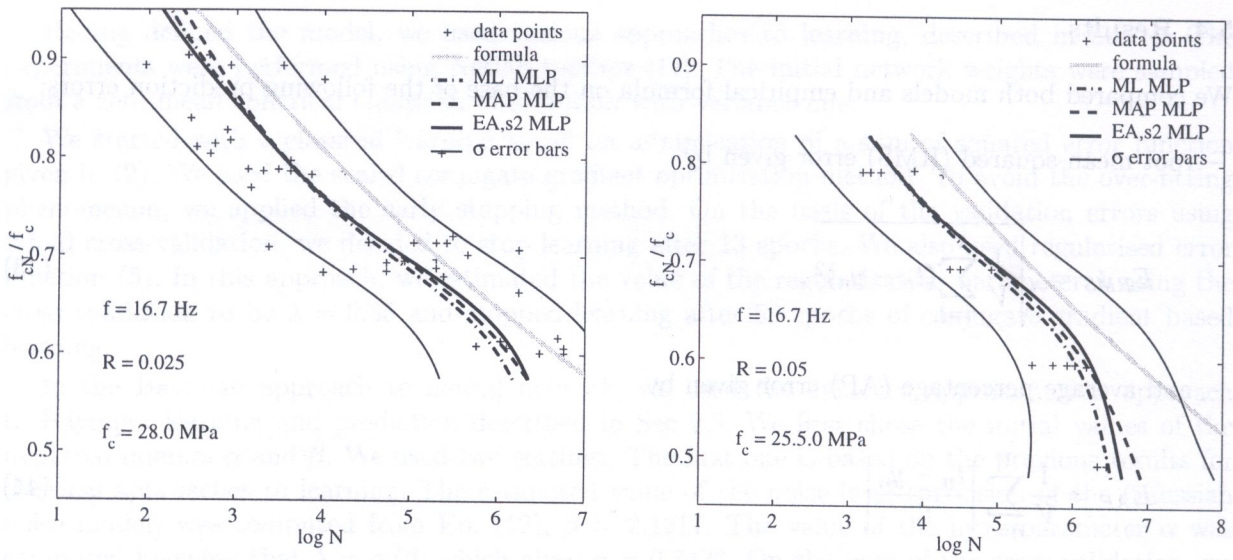

Fig. 2. Predicted by MLP models fatigue failure of concrete as function of maximal stress level for tests performed by: (left) Artim and McLaughlin, (right) Gray *et al.*

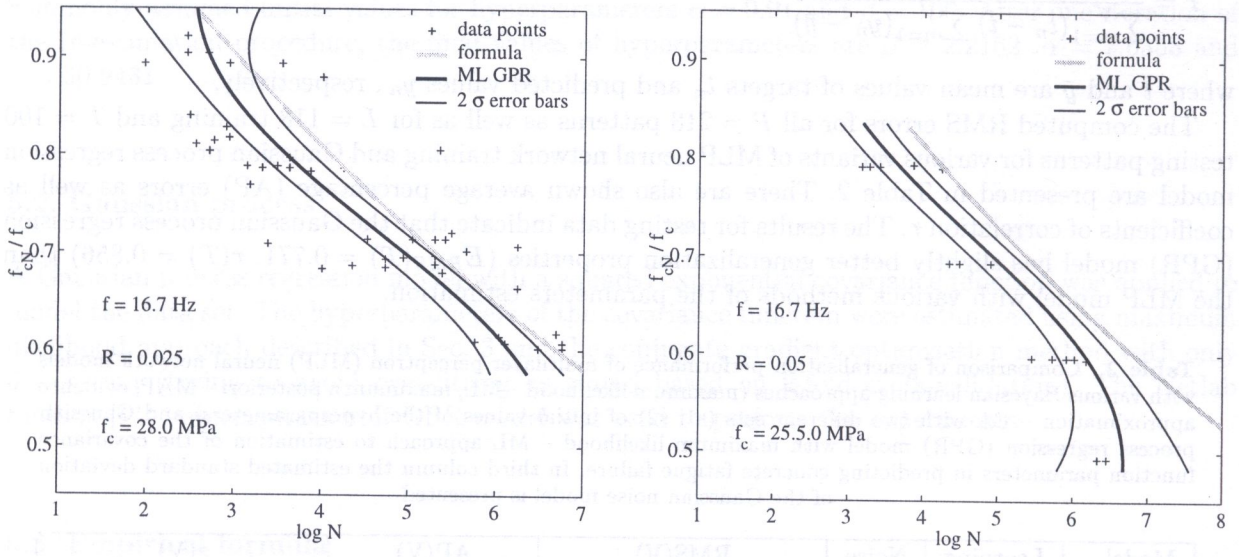

Fig. 3. Predicted by GPR model fatigue failure of concrete as function of maximal stress level for tests performed by: (left) Artim and McLaughlin, (right) Gray *et al.*

estimated values of standard deviation σ of the assumed Gaussian noise model (5) for all nonlinear regression models, which are presented in Table 2. The standard deviation values are rather similar. This may indicate that the Gaussian process model is more confident in predicting fatigue failure. For MLP models with evidence approximation, the larger error bars may be caused by much more uncertainty in estimation of parameters values (weights).

The empirical formula by Furtak is also presented in Figs. 2 and 3. It is visible that both models are closer to the experimental data than Furtak's formula, which significantly overestimates the fatigue failure.

Figure 4 shows comparisons between the measured fatigue failure and that predicted by the models for the training and the testing data. A smaller dispersion of the points around the dashed line means a smaller prediction error. MLP and GPR models seem to have very similar predictions for the experimental data.

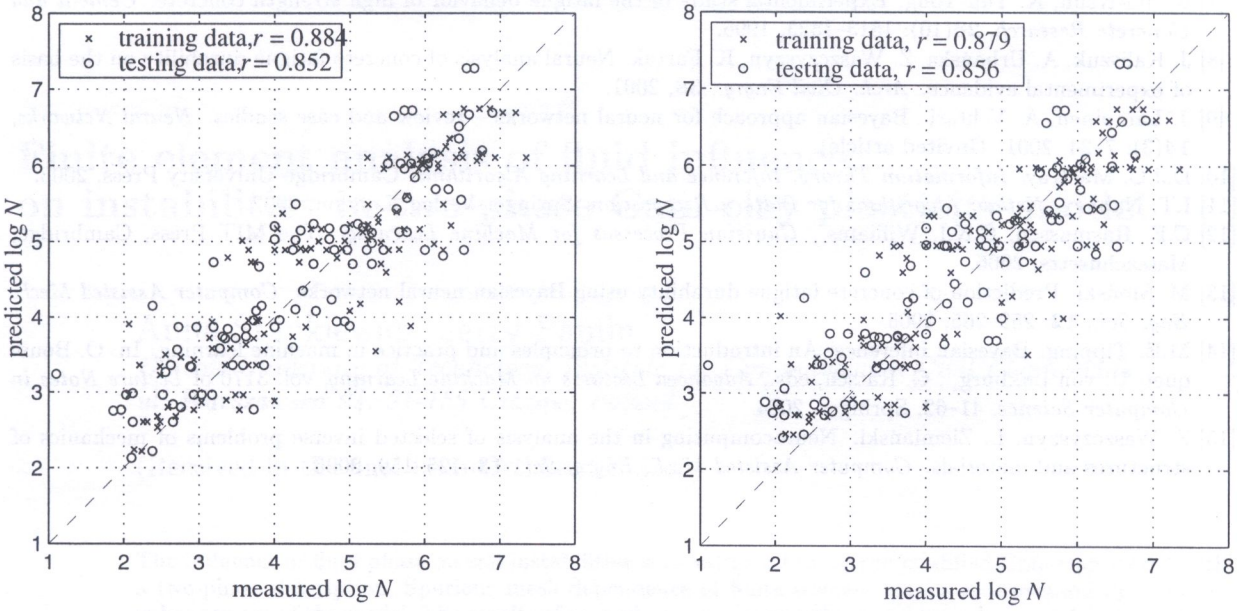

Fig. 4. Predicted vs measured fatigue failure for the training and testing patterns: (left) Bayesian neural network with evidence approximation (MLP EA, s2), (right) Gaussian process regression (GPR)

6. Conclusions

In this paper, we presented the application of Bayesian regression approach to concrete fatigue failure prediction. We used two nonlinear regression models, namely multilayer perceptron (MLP) neural network and Gaussian process regression (GPR). We applied various simple approximation approaches to full Bayesian learning and prediction.

From the results of experiments, some conclusions can be drawn. The best predictions for testing data were obtained for Gaussian process model. In the case of MLP network, the evidence approximation approach to Bayesian learning gave the smallest testing errors. Both the best models make it possible to obtain the predictive distributions and the error bars can easily be plotted.

In future, we plan to apply some improvements to the Bayesian regression approach used in this paper. It is important to consider other, more appropriate noise models. Also, we plan to use other covariance functions in GPR model. Finally, we should consider applying the stochastic approximation to Bayesian inference based on the Monte Carlo sampling method to overcome some problems with over-fitting (for the maximum likelihood based learning) in case of covariance function parameters estimation in Gaussian process regression model.

References

[1] C.A.L. Bailer-Jones, T.J. Sabin, D.J.C. MacKay, P.J. Withers. Prediction of deformed and annealed microstructures using Bayesian neural networks and Gaussian processes. In: *Proc. of the Australia-Pacific Forum on Intelligent Processing and Manufacturing of Materials*, 1997.

[2] C.M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, Oxford, 1995.

[3] C.M. Bishop. *Pattern Recognition and Machine Learning.* Springer, New York, 2006.

[4] C.M. Bishop, M.E. Tipping. Bayesian regression and classification. In: J. Suykens, G. Horvath, S. Basu, C. Micchelli, J. Vandewalle, eds., *Advances in Learning Theory: Methods, Models and Applications*, NATO Science Series III: Computer and Systems Sciences, 267–285. IOS Press, 2003.

[5] K. Furtak. Strength of the concrete under multiple repeated loads, (in Polish). *Arch. Civil Engrg.*, **30**, 1984.

[6] M. Jakubek, Z. Waszczyszyn. Neural analysis of concrete fatigue durability by the neuro-fuzzy FWNN. In: L. Rutkowski, J. Siekmann, R. Tadeusiewicz, L.A. Zadeh, eds., *Proc. Artificial Intelligence and Soft Computing ICAISC2004 – 7th Int. Conf.*, 1075–1080. T.U. of Częstochowa, Springer, Częstochowa/Zakopane, 2004.

[7] K. Jin-Keun, K. Yun-Yong. Experimental study of the fatigue behavior of high strength concrete. *Cement and Concrete Research*, **26**(10): 1513–1523, 1996.
[8] J. Kaliszuk, A. Urbańska, Z. Waszczyszyn, K. Furtak. Neural analysis of concrete fatigue durability on the basis of experimental evidence. *Arch. Civil Engrg.*, **38**, 2001.
[9] J. Lampinen, A. Vehtari. Bayesian approach for neural networks – review and case studies. *Neural Networks*, **14**(3): 7–24, 2001. (Invited article).
[10] D.J.C. MacKay. *Information Theory, Inference and Learning Algorithms.* Cambridge University Press, 2003.
[11] I.T. Nabney. *Netlab: Algorithms for Pattern Recognition.* Springer-Verlag, London, 2002.
[12] C.E. Rasmussen, C.K.I. Williams. *Gaussian Processes for Machine Learning.* The MIT Press, Cambridge, Massachusetts, 2006.
[13] M. Słoński. Prediction of concrete fatigue durability using Bayesian neural networks. *Computer Assisted Mech. Eng. Sci.*, **12**: 259–265, 2005.
[14] M.E. Tipping. Bayesian inference: An introduction to principles and practice in machine learning. In: O. Bousquet, U. von Luxburg, , G. Rätsch, eds., *Advanced Lectures on Machine Learning*, vol. 3176 of *Lecture Notes in Computer Science*, 41–62. Springer, 2004.
[15] Z. Waszczyszyn, L. Ziemiański. Neurocomputing in the analysis of selected inverse problems of mechanics of structures and materials. *Computer Assisted Mech. Engrg. Sci.*, **13**: 125–159, 2006.