http://dx.doi.org/10.7494/automat.2012.16.1.15

Jerzy Baranowski*, Waldemar Bauer*

Trust Region Based Parametric Optimisation for Nonlinear Systems

1. Introduction

In this paper we consider parametric optimisation of a controller for nonlinear system. Our focus is on systems in the form

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t)\end{aligned} \tag{1}$$

where $\mathbf{f} : \mathbb{R}^n \times \mathbb{R}^r \to \mathbb{R}^n$ is twice differentiable, $\mathbf{C} \in \mathbb{R}^{m\times n}$ with controllers in the form of generalized PI controller

$$\mathbf{u}(t) = K_p(\mathbf{w}(t) - \mathbf{y}(t)) + K_i \int_0^t (\mathbf{w}(t) - \mathbf{y}(t))\mathrm{d}t \tag{2}$$

where $K_p, K_i \in \mathbb{R}^{r\times m}$ and $\mathbf{w}(t) \in \mathbb{R}^m$ is a reference signal. We will consider optimisation of parameters K_p, K_i in order to minimise performance index in the form of

$$\begin{aligned}q(K_p, K_i) = (\mathbf{w}(T) - \mathbf{y}(T))^{\mathrm{T}} \mathbf{Q}(\mathbf{w}(T) - \mathbf{y}(T)) + \\ + \int_0^T L(\mathbf{y}(t), \mathbf{w}(t))\mathrm{d}t\end{aligned} \tag{3}$$

where $L : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}$ is also twice differentiable and $\mathbf{Q} = \mathbf{Q}^T \in \mathbb{R}^{m\times m}$ is positive definite. This class of problems is very general, and contains for example a standard PI controller optimisation, for SISO system with and integral square error criterion (ISE) with finite horizon.

* AGH University of Science and Technology, Department of Automatics, Krakow, Poland

2. State space formulation

Controllers like (2) are usually described with transfer functions. Because considered system (1) is nonlinear we will use the state space formulation. We will introduce additional m auxiliary variables $\mathbf{x}^* = [x_{n+1} \dots x_{n+m}]$ described by following system of equations

$$\dot{\mathbf{x}}^* = \mathbf{w}(t) - \mathbf{y}(t) = \mathbf{w}(t) - \mathbf{C}\mathbf{x}(t) \tag{4}$$

using these variables, we can write closed loop system as

$$\begin{aligned} \dot{\mathbf{x}}(t) = &\quad \mathbf{f}(\mathbf{x}(t), K_p(\mathbf{w}(t) - \mathbf{C}\mathbf{x}(t)) + K_i\mathbf{x}^*(t)) \\ \dot{\mathbf{x}}^*(t) = &\quad \mathbf{w}(t) - \mathbf{C}\mathbf{x}(t) \end{aligned} \tag{5}$$

Also, what will be explained later let us model parameters of controller as state variables described by

$$\dot{\mathbf{K}}_\mathbf{p}(t) = 0, \quad \dot{\mathbf{K}}_\mathbf{i}(t) = 0 \tag{6}$$

It can be seen, that $\mathbf{K_p} \equiv \mathbf{K_p}(0)$ and $\mathbf{K_i} \equiv \mathbf{K_i}(0)$ so this rewrite does not influence the control structure. Additionally we will add an auxiliary variable x_q given by

$$\dot{x}_q(t) = L(\mathbf{C}\mathbf{x}(t),\ \mathbf{w}(t)) \tag{7}$$

It can be easily seen that:

$$x_q(T) = \int_0^T L(\mathbf{C}\mathbf{x}(t),\ \mathbf{w}(t))\mathrm{d}t$$

Using this variable we can reformulate performance index as

$$\begin{aligned} q(\mathbf{K_p}, \mathbf{K_i}) = &\quad -2\mathbf{w}(T)^{\mathrm{T}}\mathbf{C}\mathbf{x}(T) + \mathbf{w}(T)^{\mathrm{T}}\mathbf{Q}\mathbf{w}(T) + \\ &+ \mathbf{x}(T)\mathbf{C}^{\mathrm{T}}\mathbf{Q}\mathbf{C}(T)\mathbf{x}(T) + x_q(T) \end{aligned} \tag{8}$$

Because $\mathbf{w}(T)^{\mathrm{T}}\mathbf{Q}\mathbf{w}(T)$ is constant it does not influence the optimisation and therefore can be omitted. Collecting all these reformulations we get a closed loop system

$$\dot{\mathbf{X}}(t) = F(\mathbf{X}(t), t) \tag{9}$$

where $\mathbf{X}(t) = [\mathbf{x}(t)\mathbf{x}^*\mathbf{K}_{\mathbf{p},11}\ldots\mathbf{K}_{\mathbf{p},r1}\ldots\ \mathbf{K}_{\mathbf{p},rm}\ \mathbf{K}_{\mathbf{i},11}\ldots\mathbf{K}_{\mathbf{i},rm}\ x_q]^{\mathrm{T}}$ and

$$F(X(t),t)) = \begin{bmatrix} f(\mathbf{x}(t), \mathbf{K}_{\mathbf{p}}(\mathbf{w}(t) - \mathbf{C}\mathbf{x}(t)) + \mathbf{K}_{\mathbf{i}}\mathbf{x}^*(t)) \\ \mathbf{w}(t) - \mathbf{C}\mathbf{x}(t) \\ \mathbf{0}_{2rm\times 1} \\ L(\mathbf{C}\mathbf{x}(t), \mathbf{w}(t)) \end{bmatrix} \tag{10}$$

with $\mathbf{X}(0)=\mathbf{X}_0 \in \mathbb{R}^{n+m+2rm+1}$ and performance index becomes

$$q(\mathbf{X}_0) = \mathbf{X}(T)^{\mathrm{T}}\mathbf{W}\mathbf{X}(T) + \mathbf{R}^{\mathrm{T}}\mathbf{X}(T) \tag{11}$$

so the parametric optimisation problem (1), (2), (3) is equivalent to the optimisation of an initial condition of nonlinear system (9) with performance index (11), and that more general problem will be considered in the following part of the paper.

3. Optimisation

In the earlier works of the authors (see [3]) the above formulation of optimisation problem was solved with the use of line search methods – namely Newton's method. Newton's method is a very quickly convergent method (quadraticaly convergent) in the convergence area. However in order to obtain such convergence Hessian has to be positive definite. It is not always the case in the context of nonlinear problems, and basic Newton's method has problems to obtain the solution (see [7]). There are of course methods that improve the convergence area, however that convergence is no longer quadratic. Also all kind of line search methods have their own problems, as they my require multiple evaluations of the performance index. Different approach for solving nonlinear optimisation problems is the trust-region problem.

4. The Trust-Region Method

We are interested in problems in the following form

$$\min_x q(x)$$

In the algorithmic strategy, known as trust region method, the information gathered about q is used to construct a model function m_k whose behavior near the current point x_k is

similar to that of the actual objective function q. Because the model m_k may not be a good approximation of q when x is far from x_k, we restrict the search for a minimiser of m_k to some region around x_k. In other words, we find the candidate step p by approximately solving the following subproblem:

$$\min m_k(\mathbf{x}_k + \mathbf{p}_k)$$

where $\mathbf{x}_k + \mathbf{p}_k$ lies inside the trust region.

If the candidate solution does not produce a sufficient decrease in q, we conclude that the trust region is too large, and we reduce it and re-solve the subproblem. Usually, the trust region is a ball defined by $\|\mathbf{p}_k\| = \Delta_k$, where the scalar $\Delta_k > 0$ is called the trust-region radius - norm used is usually Euclidean, however also scaled euclidean, taxicab and maximum norms are used. Sometimes there are changed from one iteration to another.

The model m_k in subproblem is usually defined to be a quadratic function of the form

$$m_k(\mathbf{x}_k + \mathbf{p}_k) = q_k + \mathbf{p}_k^{\mathrm{T}} \nabla_x \mathbf{q}_k + \frac{1}{2} \mathbf{p}_k^{\mathrm{T}} \nabla_{xx} q_k \mathbf{p}_k$$

where q_k, $\nabla_x q_k$ and $\nabla_{xx} q_k$ are a function value at x_k, gradient of function at x_k, and Hessian matrix of function at x_k, respectively (for details see [4, 7]).

This concept allowed for the creation of Basic trust-region algorithm (BTR):

Initialization:

An initial point x_0, intial value of trust-region radius Δ_0 , and maximal radius $\hat{\Delta}$ are given. Also the constants η_1, η_2 such that

$$0 < \eta_1 \leq \eta_2 < 1 \tag{12}$$

are given. Compute initial value of performance index $q(x_0)$ and set $k = 0$.

Step 1: Model definition

Choose the norm and construct the model m_k in the trust-region.

Step 2: Step calculation

Compute the step $\mathbf{p}_k$ that reduces the model and such that $\mathbf{x}_k + \mathbf{p}_k$ belong to trust-region. Usually by minimisation of the model over region or by approximate minimisation.

Step 3: Compare the reduction of the model vs reduction of the function

$$\rho_i = \frac{q(\mathbf{x}_k) - q(\mathbf{x}_k + \mathbf{p}_k)}{m_k(\mathbf{x}_k) - m_k(\mathbf{x}_k + \mathbf{p}_k)}$$

If $\rho_k = \eta_1$, then $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{p}_k$, otherwise $\mathbf{x}_{k+1} = \mathbf{x}_k$.

Step 4: Update radius

Choose Δ_{k+1} from appropriate interval:

$$\Delta_{k+1} \in \begin{cases} [\Delta_k, \hat{\Delta}) & \text{if} \quad \rho_k \geq \eta_2, \\ [\eta_2 \Delta_k, \Delta_k] & \text{if} \quad \rho_k \in [\eta_1, \eta_2), \\ [\eta_1 \Delta_k, \eta_2 \Delta_k] & \text{if} \quad \rho_k < \eta_1 \end{cases}$$

$k = k + 1$ and go to Step 1.

4.1. Solving the trust-region subproblem

To find the next trial point we need to formulate and solve the Trust-Region subproblem. Usually depending on number of the variables solutions are obtained exactly or approximately. Usually if there are no more than about 50 variables exact methods canbe used – otherwise such methods as Cauchy point, dog-leg and subspace minimisation are used.

In this paper for the solution of the subproblem an exact method was used. The global solution to the subproblem is given by the following.

Theorem 1. (see [7]) The vector $\mathbf{p}^*$ is a global solution of the trust-region subproblem

$$\min_{\mathbf{p}_k \in \Re^n} q_k + \mathbf{p}_k^{\mathrm{T}} \mathbf{g}_k + \frac{1}{2} \mathbf{p}_k^{\mathrm{T}} \mathbf{H}_k \mathbf{p}_k \quad s.t. \quad \|\mathbf{p}_k\| < \Delta_k$$

if and only if $\mathbf{p}^*$ is feasible and there is a scalar $\lambda \geq 0$ such that the following conditions are satisfied:

$$(\mathbf{H}_k + \lambda \mathbf{I})\mathbf{p}^* = -\mathbf{g}_\mathbf{k}$$

$$\lambda(\Delta_k - \|\mathbf{p}^*\|) = 0$$

$$(\mathbf{H}_k + \lambda \mathbf{I}) \geq 0$$

In this paper – because of the low dimension of the considered problems we have used the following strategy:

1) Verify if $\mathbf{H}_k \geq 0$ if so, solve:

$$\mathbf{H}_k \tilde{\mathbf{p}} = -\mathbf{g_k}$$

and check if $\Delta_k \geq \|\tilde{\mathbf{p}}\|$. If this condition is fulfilled set $\mathbf{p}^* = \tilde{\mathbf{p}}$. This situation is a Newton's step of length $\leq \Delta_k$

2) Otherwise, find solution $\mathbf{p}(\lambda)$ such that

$$(\mathbf{H}_k + \lambda \mathbf{I})\mathbf{p}(\lambda) = -\mathbf{g_k}$$

for low dimensional problems Cramer formula can be used.

3) Find roots of the following polynomial

$$w(\lambda) = (\mathbf{p}(\lambda)^{\mathrm{T}}\mathbf{p}(\lambda) - \Delta_k^2)\det(\mathbf{H}_k + \lambda \mathbf{I})$$

These operations can be easily computed with Matlab.

4) Choose a root λ^* of $\mathbf{w}(\lambda)$ which fulfills

$$(\mathbf{H}_k + \lambda \mathbf{I}) \geq 0$$

and set $\mathbf{p}^* = \mathbf{p}(\lambda^*)$.

In problems of higher dimension certain iterative schemes must be used (see [4, 7]). In order to use trust-region algorithm we need to compute gradient and Hessian which are given in the following section.

5. Gradients with respect to initial conditions

Our analysis will be based on the so called *variational equation* (see. [6] p. 95). Variational equation takes form

$$\dot{\mathbf{\Phi}}(t) = \mathbf{J}(\mathbf{X}(t))\mathbf{\Phi}(t) \tag{13}$$

$$\mathbf{\Phi}(0) = \mathbf{I}_{N\times N} \tag{14}$$

where $\mathbf{J}(\mathbf{X}(t)) \in \mathbb{R}^{N\times N}$ is a Jacobi matrix of right hand side of (9). Variational equation has a following property

$$\frac{\mathrm{d}\mathbf{X}(t)}{\mathrm{d}\mathbf{X}_0} = \mathbf{\Phi}(t) \tag{15}$$

It should be noted, that variational equation(13) is linear time varying equation, that can be solved simultaneously with (9). We will now use the variational equation to compute the gradient of $q(\mathbf{X}_0)$ that is

$$\nabla_{X_0} q(\mathbf{X}_0) = \mathbf{\Phi}(T)^{\mathrm{T}}(2\mathbf{W}\mathbf{X}(T) + \mathbf{R}) \tag{16}$$

Conclusion, is that in order to obtain gradient of (11) with respect to initial condition $\mathbf{X}_0$ needs to solve a system of N^2 differential equations (13).

6. Hessian with respect to initial conditions

The Hessian $\nabla_{x0\,x0}\, q(\mathbf{X}_0) = \mathbf{H}(\mathrm{T})$ is given by

$$\mathbf{H}(T) = \mathbf{V}(T) + 2\mathbf{\Phi}(T)^{\mathrm{T}}\mathbf{W}\mathbf{\Phi}(T) \tag{17}$$

$$\mathbf{V}(t) = \begin{bmatrix} (\mathbf{V}_1(t)(2\mathbf{W}\mathbf{X}(T) + \mathbf{R}))^{\mathrm{T}} \\ \vdots \\ (\mathbf{V}_N(t)(2\mathbf{W}\mathbf{X}(T) + \mathbf{R}))^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \tag{18}$$

$$\dot{\mathbf{V}}_j(t) = \sum_{p=1}^{N} (\phi_{pj}(t) \frac{\partial}{\partial z_p} \mathbf{J}(z)\,|_{z=X(t)}) \mathbf{\Phi}(t) + \mathbf{J}(\mathbf{X}(t))\mathbf{V}_j(t) \tag{19}$$

$$\dot{\mathbf{\Phi}}(t) = \mathbf{J}(\mathbf{X}(t))\mathbf{\Phi}(t) \tag{20}$$

$$\mathbf{\Phi}(0) = \mathbf{I}_{N\times N} \tag{21}$$

$$\mathbf{V}_j(0) = \mathbf{0}_{N\times N},\ j = 1,2,...N \tag{22}$$

Because $\mathbf{\Phi}$ was computed already in order to determine the gradient, then the effective numerical cost of Hessian computation is solving N^3 differential equations. For determination of Hessian for this performance index see [3], and for more detailed discussion of derivatives with respect to initial conditions see [2]. It should be noted, that both gradient and Hessian computation gives us derivatives with respect to all initial conditions. To obtain needed derivatives with respect to parameters one needs to extract appropriate rows and columns.

7. Example

Let us consider a parametric optimisation of series DC motor with PI speed controller. Model of the motor (see [1, 5]) is given by

$$\begin{aligned} \dot{x}_1(t) = &\quad -\alpha_1 x_1(t) - \alpha_2 x_1(t) x_2(t) + \beta u(t) \\ \dot{x}_2(t) = &\quad \gamma_1 x_1(t)^2 - \gamma_2 x_2(t) - \tau(t) \end{aligned} \tag{23}$$

where τ is load torque, x_1 is motor current and x_2 is angular velocity. Controller has form

$$u(t) = K_p(w(t) - x_2(t)) + K_i \int_0^t (w(t) - x_2(t)) \mathrm{d}t \tag{24}$$

We want to minimise the performance index

$$q(K_p, K_i) = \int_0^T (w(t) - x_2(t))^2 \mathrm{d}t \tag{25}$$

This problem is equivalent to system (9) where

$$X_1(t) = \int_0^t (w(t) - x_2(t)) \mathrm{d}t, \quad X_6(t) = \int_0^t (w(t) - x_2(t))^2 \mathrm{d}t,$$

$$X_2(t) = x_1(t),\ X_3(t) = x_2(t),\ X_4(t) = K_p,\ X_5(t) = K_i$$

Equations of right hand side of differential equations are given in [3]. The performance index takes form

$$q(X_0) = X_6(T) \tag{26}$$

that means, that matrices of performance index (11) are $\mathbf{W} = \mathbf{0}_{6\times 6}$ and $\mathbf{R} = \mathbf{e_6}$ (with $\mathbf{e}_i$ denoting the i-th unit vector). We can apply then formula for gradient (16), and the Hessian matrix can be computed with formulas (17)–(24). Analytical computation of coefficients for this formula see [3].

For optimisation a trust region algorithm was used where $\eta_1 = 0.25$ and $\eta_2 = 0.75$ (see for example [7]). Algorithm was stopped when gradient $\mathbf{g}$ of performance index with respect to optimised parameters would fulfil the condition $\| \mathbf{g} \| < 10^{-10}$. For computation of gradients a multi-step backward difference method of Klopfenstein-Shampine family was used. It is implemented in Matlab as `ode15s` [8]. Absolute and relative tolerance of the method were set both equal to 10^{-9}. Algorithm determined a minimum from multiple (stable

and unstable) starting values in less than 20 iterations. Step response of the system with an example of initial and optimised parameters is presented in the Figure 1. Iterations of algorithm with trust regions are presented in the Figure 2.

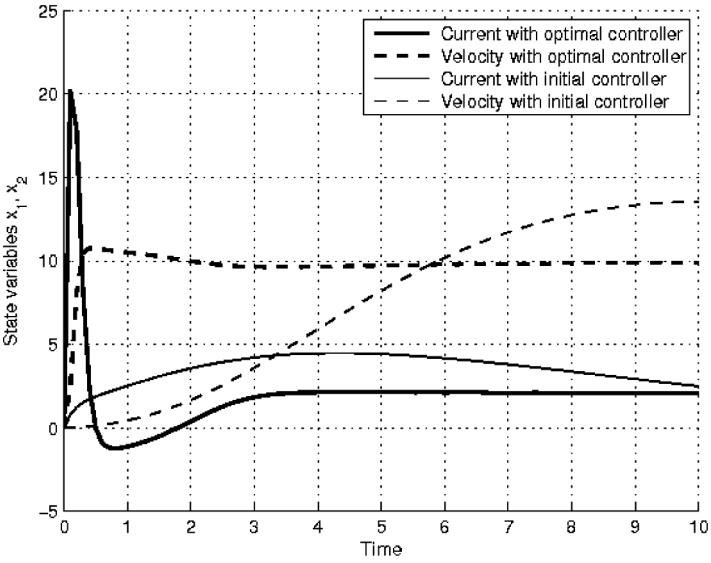

Fig. 1. State variables of the motor with optimised and initial

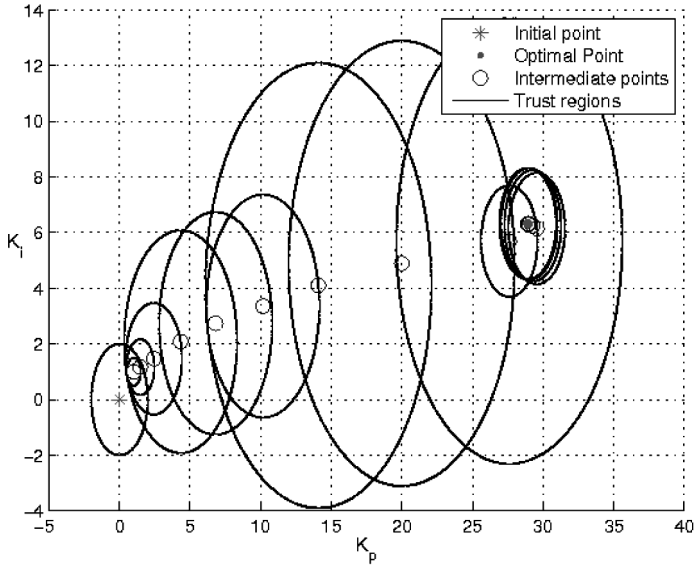

Fig. 2. Iterations in parameter space along with trust regions

8. Conclusions

In the paper a trust region method was successfully adapted to the task of parametric optimisation. Method of analytical computation of derivatives presented in [3] gave satisfactory results quickly finding optimum.

Acknowledgements

Work partially financed from statutory funds contract no. 11.11.120.768 and partially from NCN-National Science Centre funds no. N N514 644440.

References

[1] Baranowski J., *Projektowanie obserwatora dla silnika szeregowego prądu stałego*. Automatyka (półrocznik AGH), 10, 1, 2006, 33–52.
[2] Baranowski J., *Odtwarzanie stanu systemów dynamicznych z dyskretnych danych pomiarowych*. PhD thesis, Akademia Górniczo-Hutnicza im. Stanisława Staszica, Wydział Elektrotechniki, Automatyki, Informatyki i Elektroniki, Kraków, 2010.
[3] Baranowski J., Długosz M., Mitkowski W., *Parametric optimization of nonlinear system controller*. [in:] Materiały XIII sympozjum Podstawowe Problemy Energoelektroniki, Elektromechaniki i Mechatroniki, Wisła, 14–17 grudnia 2009, 206–211.
[4] Conn A.R., Gould N.I.M., Toint P.L., *Trust-Region Methods*. MPS/SIAM, Philadelphia, 2000.
[5] Dlugosz M., *Problemy optymalizacji układów napędowych w automatyce i robotyce*. PhD thesis, Akademia Górniczo-Hutnicza, 2009. Supervisor: W. Mitkowski.
[6] Hairer E., Nørsett S., Wanner G., *Solving Ordinary Differential Equations: I Nonstiff problems*. 2nd ed., Springer, 2000.
[7] Nocedal J., Wright S., *Numerical Optimization*. 2nd edition, Springer, 2006.
[8] Shampine L.F., Gladwell I., Thompson S., *Solving ODEs with MATLAB*. Cambridge University Press, 2003.