DIFFERENTIATION, SENSITIVITY ANALYSIS AND IDENTIFICATION OF HYBRID MODELS

Symbolic derivative of Simulink©models

John Masse
Appedge, 18-22 rue d'Arras
92000 Nanterre
john.masse@qppedge.com, http://www.appedge.com

Thierry Cambois
PSA Peugeot Citroen, 18 rue des Fauvelles, 92250 la Garenne-colombes
thierry.cambois@mpsa.com

Abstract

Diffedge tool is a new methodology that eliminates the drawbacks of finite-difference approximations and the complexity to use the automatic differentiation. It combines the powerful of computer algebra system and block diagram structures for computing the derivative of a Simulink model with respect to the independent parameters. *Diffedge* calculates the symbolic derivative of the mathematical models described in the form of block diagram by the application of JM's rules. The obtained symbolic derivative is also represented in graphic form (block diagrams) and can be used like any Simulink model. The benefits for the user are numerous. We are remaining in the Simulink environment. *Diffedge* does not require any task of programming and modification of the model. The visualisation of the partial derivatives is possible for any coordinates of the model. This presentation illustrates the capabilities of *Diffedge* , its implementation in the Matlab©environment and some applications.

Keywords: Simulink differentiation, optimisation, identification protocol, parametric sensitivity and statistical analysis, *Diffedge*

Introduction

Although numerical simulation is more and more used to predict operation of practical systems, its development remains one of the main

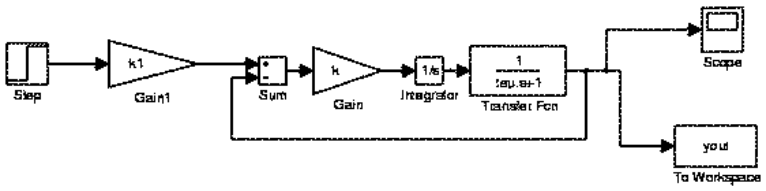

Figure 1. Closed-loop model

challenges of the next coming years. In many fields, simulation will reduce drastically design development cost, for instance, by enabling to build digital mock-up of systems in order to:

- Understand static or dynamic behaviour,
- Conduct fault tolerance tests for critical applications or for real time when only a "virtual" design is available,
- Prepare test campaigns for real hardware,
- Conduct parametric studies in order to choose an optimal design.

There are several possibilities to build a mathematical model. The most popular of CACSD software is Matlab with its blocks diagrams GUI (Simulink), covering the field of automatic control and optimisation. However, in many situations the size and complexity (hybrid system, discontinuities, logical evens, strongly non-linear...) of the resulting block diagrams make it difficult to apply computational technicals for identification and optimal control. This is because the numerical gradient calculation of the optimizing function with respect to independent parameters is in many cases inaccurate or wrong and unreliable because of the truncation or cancellation errors.

1. Possibilities of *Diffedge*

In these situations it is preferable to rely on *Diffedge*. *Diffedge* is a new methodology that eliminates the drawbacks of finite difference approximations and the complexity of the automatic differentiation approach. It combines the powerful of computer algebra system and block diagram structures for computing the derivative of a Simulink model with respect to the model parameters.

Diffedge calculates the symbolic derivative of the Simulink model described in the form of block diagram (figure 1) by the application of JM's rules(*Diffedge* differentiation rules will be described in forthcoming work). The obtained symbolic derivative is also represented in graphic

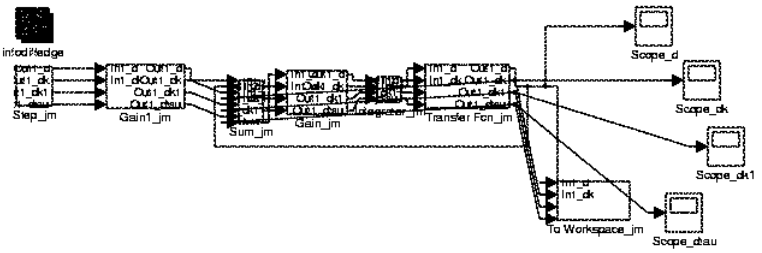

Figure 2. First derivative of Closed-loop model with respect to $k, k1, tau$

form (block diagrams figure 2) and can be used like any Simulink model and so; all functionalities of Simulink/Matlab can be used on the derivative models such as RTW©.

The benefits for the user are numerous:

- By remaining in the environment of simulation, it is possible to obtain the second or higher order derivatives by applying *Diffedge* again on the derivative model(figure 3) .
- No additional programming is required and modification of the model just necessitates the name of the mdl file and the list of the parameters for which, one wishes to obtain the derived model.
- The visualisation of the partial derivatives is possible for any coordinate of the model (see figure 8).

This presentation illustrates the capabilities of *Diffedge* , its implementation in the Matlab environment. The fields cover by *Diffedge* are the followings:

- Analytic sensitivity and statistical analysis
- Optimisation and identification
- Fault detection for system monitoring in real-time
- Calibrating and validating model (parameters and structure)

2. *Diffedge* and the toolboxes Matlab

Diffedge does not include the Extend Symbolic Math toolbox and the Optimization toolbox of Matlab.

The computer algebra system is used for computing the symbolic derivatives with respect to parameters. A large choice of optimisation

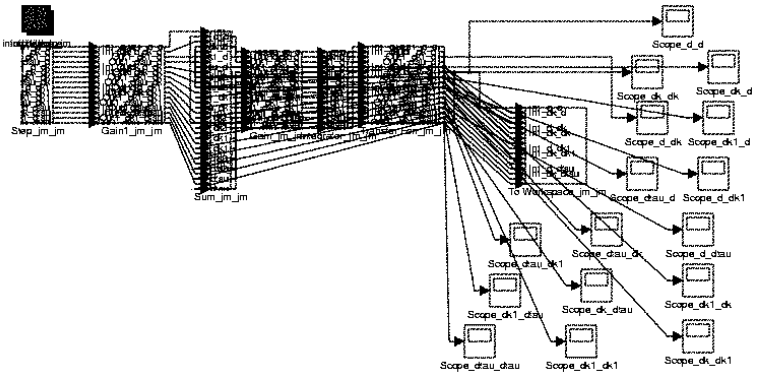

Figure 3. Second derivative of Closed-loop model with respect to $k, k1, tau$

routines allows to try several algorithms. Consequently you can choose the optimisation routines the most suitable for your model. This is useful if you need to ensure that you use the best algorithm. In fact, like the ODE solvers, no single method of optimisation suffices for all systems. *Diffedge* provides the symbolic gradient, jacobian or hessian of your model and embeds your optimisation routines. *Diffedge* is an automatic tool for optimisation problems.

3. *Diffedge* and the integration strategy.

The mathematical models often contain discontinuities or nonlinear blocks such as saturated integrator, switch, min, max, enable, trigger, ... Simulink uses the zero crossing detection technique for checking and detecting events or discontinuities. The advantage of *Diffedge* is to remain in the same environment; we have access at the same functionality

This potentiality is interesting when you want to make parametric sensitivity analysis in order to obtain information concerning parameter variations in our hybrid model, for the most difficult point is to compute the sensitivity function $S(t,p) = \frac{\partial Y(t)}{\partial p}$ and p denotes the parameter and Y the model output.

The parametric sensitivity analysis will be unreliable if we don't use a satisfactory strategy integration and a symbolic derivative. Due to the fact the sensitivities will often jump at the discontinuities, if the numerical integration is not stopped and the jump computed explicitly, the computed sensitivity trajectories will generally be incorrect (quantitatively and qualitatively). And also computing finite difference derivative is not a good way when our model has discontinuities or noise, because we cumulate the error in the finite difference of sensitivities due to the

truncation error because the ODE integration is performed with limited tolerance integration (*rtol* and *atol*). Another advantage of *Diffedge* is to compute all the sensitivity trajectories in one time. A numerical derivative requires, in order to compute the sensitivity functions with the respect to np parameters, $np + 1$ numerical integrations. Also, the user must ensure that the integration step size (for fixed step solvers), selected for each simulation is appropriate for $np+1$ integrations and the δp of each parameter is also appropriate. That is more complicated when we use optimisation algorithm because the parameter values change for each optimisation iteration. Furthermore the partial derivative computing with finite differences $S(t,p) = \frac{Y(t,p+\delta p)-Y(t,p)}{\delta p}$ is valid as long as δp is sufficiently small for producing substantially the same behaviour of the simulation and it must not be too small with respect to integration tolerances otherwise the derivative is zero. In some cases, the errors done by finite differences can introduce unacceptable inaccuracies in$S(t,p)$ .The use of symbolic derivative avoids all these drawbacks.

Diffedge allows to perform an accurate parametric sensitivity analysis of system containing discontinuities and it provides an automatic script that the user can apply to Simulink models in order to perform correctly a sensitivity analysis more easily.

4. *Diffedge* and analytic sensitivity analysis

The main use of parametric sensibility analysis of *Diffedge* is to localize the instants where each parameter produces the maximum influence on outputs of our model. This knowledge is crucial when we want to know the behaviour of the model and to predict the effect of each parameter. The sensitivity analysis with the symbolic derivative can be an essential help to find the optimal parameter values. *This sensitivity reflects the reliability of the mathematical model for the selected trajectory with this tool, no information can escape the modeller's attention about of the structure of its mathematical model.* The symbolic statistical routine of *Diffedge* allows, under certain conditions, to expand each output concerning the nominal solution in Taylor series and truncated after the first-order or second-order terms. *Diffedge* assume a Normal distribution for each parameter and for each instant of the sensitivity trajectories. Thus, we obtain $\mathcal{L}(P_i) = \mathcal{N}(m_{p_i}, \sigma_{p_i}) \longrightarrow L(Y(t)) = \mathcal{N}(m(t)_{y_p i}, \sigma(t)_{yp_i})$with ${\sigma_{p_i}}^2$ the variance and m_{p_i} the mean value (or nominal value). *Diffedge* computes σ_{yp_i} (Outputs RMS vector), with only one simulation, and plots the RMS response over the model response (Figure 2). Thus, at each instant, we obtain a confidence interval. The maximum deviation shows the maximum sensitivity of studied

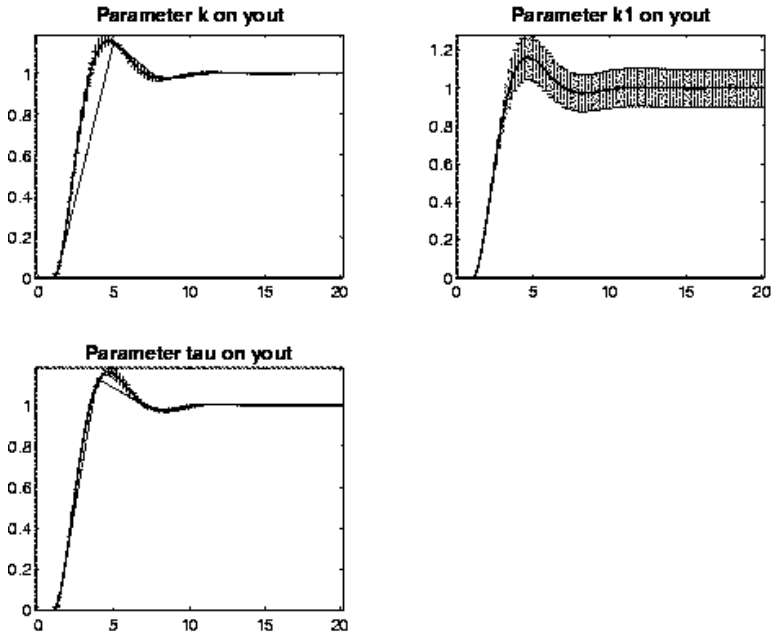

Figure 4. Sensitivity response of parameters on the ouput

parameters. We can manage the resulting RMS response. (For example we can sort by descending order the sensitivity of the parameters on the outputs (table1)). When $\sigma_{yp_i} = 0$, the parameter p_i has no sensitivity on the response $Y(t)$.

4.1 An academic example

Our academic example is the following (Figure 1). The sensitivity analysis method allows us to retrieve the following well-known property of PI controller. The integral action in the feedback is responsible for driving error to zero. Automatic control predicts that:

- The gain k into the feedback has a limited influence during the transient response like tau,
- The parameter $k1$ outside the feedback has just an action on the set point.

The resulting closed-loop step response is shown (Figure 4): We do find the localized actions of each parameter on the response like we have foreseen hereabove.

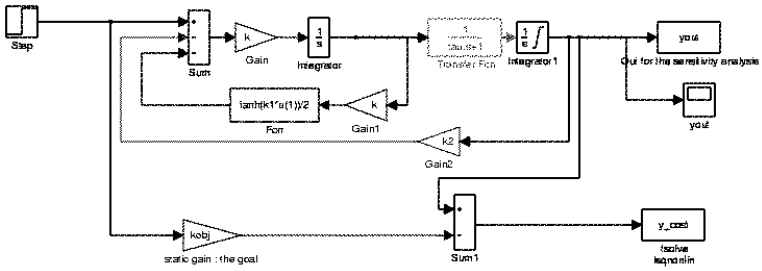

Figure 5. Non linear model with graphic cost

For the sensitivity analysis, we assume a variation of ±10% for all parameters. The table 1 gives for each parameter the RMS max and the time when it occurs on the output. We can trust in these values because the residue (Truncation error of the RMS) is small.

yout order	parameters	mean	rms max	variation in %	rms error (residue)	$\frac{dy}{dk}$ max	inst. sec
1	cumulate sensitivity $k, k1, tau$		0.120619	10.7277	0.0072398	0.43525	4
2	$k1$	1	0.11629	9.99999	0	0.20802	4.6
3	k	1	0.060052	6.47287	0.002759	0.60052	3.2
4	tau	1	0.028210	4.25938	0.000645	0.53384	2.6

Table 1. Maximum Y_{p_i} RMS values sorted by descending order

On the table we see that a variation of 10% of $k1$ gives 10% (max) of output variation at 4.6 seconds with a residue on the order of 0%. This result confirms that the parameter is defining directly the set point.

This example has shown how with the symbolic derivative (figure 2 and 3) we can easily look for the effect and the localization action of each parameter on the output of our model. When we work on large model, this method is very useful and efficient for driving and is defining precisely the best zone for computing the objective function.

5. Optimisation problem illustrating the *Diffedge* methodology

Here, we adopt two points of view for optimising a Simulink model. The first optimisation uses the classical finite-difference derivatives for

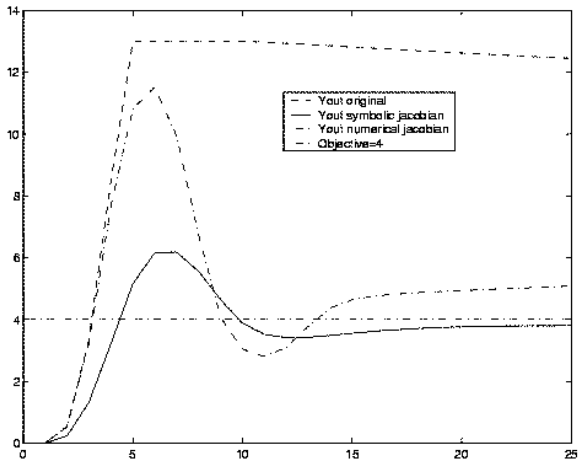

Figure 6. Compare optimization with a symbolic and numerical jacobian

computing the jacobian of the objective or constraints; the second uses a symbolic jacobian computed by *Diffedge*.

Let's say that you want to optimise the control parameters in the simulink model (Figure 5)

The model includes a discontinuity (saturated integrator) and a strong nonlinear function in the loop. The model output is yout and the objective function is y_cost. The variables k, $k1$, $k2$ are the parameters we are optimising.

The academic problem is to design a feedback control law or parametric identification that tracks our trajectory. Our goal in this problem is to minimize the error between the output and the objective signal $kobj = 4$ and $Cost = \int (yout - kobj)^2 dt$.

We choose a variable-step solver (ODE15s) for solving the stiff system (ODE). The integration is very fast because the solver follows the dynamic of the model's states.

Remark: Choosing a fixed step is not efficient and is consuming much time. The fixed step has to be small when the dynamic of model's state is high and we loose our time when the dynamic decreases.

We compute the derivative model (figure 7) with the following syntax:

$$Diffedge('derivative', NameofMymdl, \{'k','k1','k2'\})$$

The figure 6 compares the numerical optimisation with the symbolic one (symbolic jacobian) for the same optimisation routine and the same optimisation parameter set-up. The two solutions do not converge to the steady state.

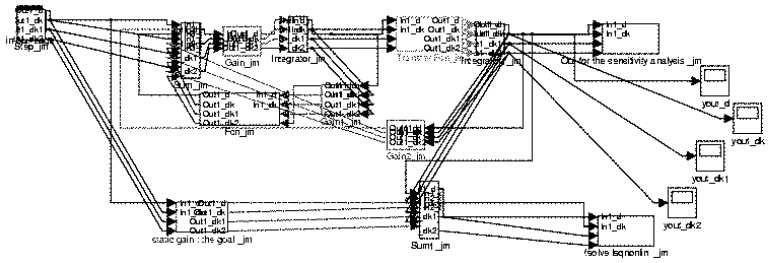

Figure 7. Result of *Diffedge* : Derivative model with respect to $k, k1, k2$

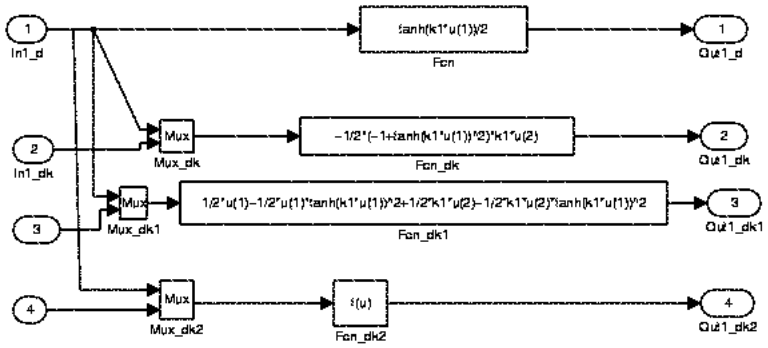

Figure 8. Zoom of Fcn : Symbolic derivative of block Fcn : $tanh(k1 * u)/2$

The symbolic gradient is more efficient, the minimization of error is better. This is due to the numerical derivative which is difficult to compute and is unreliable because the choice of (for $k, k1, k2$) is very difficult in this case and the search direction may be wrong or too weak. The optimisation algorithm stops before the end of the minimization.

6. Conclusion

Diffedge's tool with its analytic statistic toolbox covers many domains. This new tool brings another way to study hybrid models. *Diffedge* increases flexibility in mathematical information management and also end-user benefits. The use of this kind of software will probably increase during the coming years as it allows research teams to test easily new sensitivity methods or optimisation algorithms and it facilitates transfer to industrial users.

There are numerous benefits: Reduced product cost and mainly easier innovation, etc.

Diffedge has already demonstrated its ability to apply an industrial example with success: The PSA model contains more than 600 blocks (like sign, abs, trigger, enable, look-up table,...) and more than 70 parameters. All the optimisation methods used previously with numerical gradient or without derivative do not always work with efficiency. *Diffedge* allows to explore process and gain more knowledge of very complicated models. *Diffedge* becomes now an indispensable tool for studying block-diagram models.

References

[1] Gilbert J.C.. Le Vey G. and J. Masse. La diffrentiation automatique de fonctions reprsentes par des programmes. Technical report, 1991.

[2] P.J Gawthrop and E Ronco. Symbolic quasi-newton optimisation for system identification. *Control*, 2000.

[3] J. Masse. Diffedge user manual 2003 v2. Technical report, 2000.

[4] Alexandre Sedoglavic. A probabilistic algorithm to test local algebraic observability in polynomial time. *ISAAC*, 2001.