PWATOOL User Manual

Mohsen Zamani, Behzad Samadi, Luis Rodrigues

Department of Electrical and Computer Engineering, Concordia University, Montréal, Canada

Corresponding author: Luis Rodrigues, Email: luisrod@encs.concordia.ca

Contents

1 PWATOOL, The BASICS **2**
- 1.1 What PWATOOL does . . . 2
- 1.2 Installation . . . 2
- 1.3 Background . . . 2
  - 1.3.1 Defining A PWA Model . . . 2
  - 1.3.2 Defining a PWADI model . . . 3
  - 1.3.3 Region Construction In PWA /PWADI Modeling . . . 3
- 1.4 Variable Structure and Definition In PWATOOL . . . 4
  - 1.4.1 Nonlinear model; Defining 'A' and 'a' . . . 4
  - 1.4.2 Nonlinear Model; Defining $f(x)$ . . . 4
  - 1.4.3 Nonlinear model; Defining B(x) . . . 5
  - 1.4.4 Nonlinear Model; Defining The Domains of The Regions . . . 5
  - 1.4.5 Nonlinear Model; Marking the variables which appear in $f(x)$ . . . 5
  - 1.4.6 Nonlinear model; Gridding And The Number of The Regions . . . 5
  - 1.4.7 Nonlinear Model; Defining 'Rstar' Region . . . 6
  - 1.4.8 PWA Model; Defining The Dynamics . . . 6
  - 1.4.9 PWA Model; Defining The Regions R_i . . . 6
- 1.5 Aggregated Vector Space $y = [x; 1]$ . . . 6
  - 1.5.1 Aggregated Vector Space; 'Abar' . . . 6
  - 1.5.2 Aggregated Vector Space; 'Bbar' . . . 6
  - 1.5.3 Aggregated Vector Space; 'Ebar' . . . 7
  - 1.5.4 Aggregated Vector Space; 'Fbar' . . . 7
  - 1.5.5 Aggregated Vector Space; 'Kbar' . . . 7

2 Analysis of The PWA and PWADI Models; The Theory **8**
- 2.1 Global Quadratic Analysis; Nonlinear Models . . . 8
- 2.2 PWQ Analysis; Nonlinear Models . . . 10
- 2.3 Analyzing A PWA Model . . . 12

3 Synthesis of PWA Controllers for The PWA and PWADI Models; The Theory **13**
- 3.1 Global Quadratic Synthesis; Nonlinear Models . . . 13
- 3.2 PWQ Synthesis; Nonlinear Models . . . 15
- 3.3 Synthesis PWA controllers for A PWA Model . . . 17

4 Examples; Analysis and Synthesis **18**
- 4.1 Nonlinear Resistor Example; Modeling, Analysis and Synthesis . . . 18
  - 4.1.1 Creating a nonlinear model . . . 19
  - 4.1.2 Stability Analysis Of The Open-loop System . . . 22
  - 4.1.3 Synthesis Of PWA Controllers . . . 23
  - 4.1.4 Stability Analysis Of The Closed-loop System . . . 25

Chapter 1

PWATOOL, The BASICS

This is a user manual for PWATOOL which is a free, third-party MATLAB toolbox for the study of PWA systems. PWATOOL starts by importing a nonlinear or piecewise-affine (PWA) model. If the model is nonlinear, PWATOOL approximates it with Piecewise-affine Differential Inclusions (PWADI). When a PWA or PWADI model has been created, PWATOOL can analyze it for stability or synthesize PWA controllers for it, using Lyapunov function methodology. PWATOOL has a user interface and also includes several functions for creating and analyzing the PWA and PWADI models and synthesizing PWA controller for them. PWATOOL can be used for analysis and synthesis of small- to medium-sized models. It is based on solving feasibility or optimization problems subject to LMI or BMI constraints. PWATOOL uses Yalmip to solve the problems which verifies the sufficient conditions for the stability of the models or synthesis of PWA controllers.

1.1 What PWATOOL does

PWATOOL imports PWA or nonlinear models in a format that it recognizes and approximates the nonlinear models with PWADIs. PWATOOL can analyze the PWA or PWADI model around a given point to see if the system is stable around a given equilibrium point. PWATOOL either identifies the system as stable or states that it has not been able to verify the stability. PWATOOL can also design stabilizing PWA controllers for the system with different Lyapunov-based methods. Likewise the analysis, there is always a probability that PWATOOL does not converge. Changing the synthesis parameters may help in that case and therefore one or more methods may converge. If synthesis is successful, controller gains are produced and the closed-loop is simulated by using the achieved PWA controller.

Sliding modes designs are not included in PWATOOL. Therefore, PWATOOL is not able to synthesize PWA controllers for discontinuous dynamics at this time. It does not also declare a system as unstable; it can only state whether a given system is stable.

1.2 Installation

To install PWATOOL unzip the PWATOOL.zip file in the desired path and add the directory and its subdirectories to the MATLAB path cache. This is done in MATLAB by choosing the menus File $--$ >, Set Path $--$ >, Add with Subfolders, .. and then typing the name of PWATOOL main directory.

1.3 Background

In this section we explain the terms which we use in this manual. An advanced user, familiar with PWA topic, may skip this section and go directly to the next chapters. However, knowing the notation we use, is necessary in understanding this manual fully.

1.3.1 Defining A PWA Model

A piecewise-affine model is described by NR set of linear dynamics, including an affine term, in NR regions R_i $(i = 1, 2, ..., NR)$ as follows:

$$\frac{dx}{dt} = A_i x + a_i + B_i u \qquad \text{if } x \in R_i \tag{1.1}$$

where A_i is the linear dynamics, a_i the affine term and B_i the input gain for $i = 1, 2, ..., NR$. Technically, we store the dynamics data of a PWA model in a structure with array cells fields. For example if the model is called pwasys, then there exist three fields 'A', 'a' and 'B', all cells by sizes of NR x 1, so that $pwasys.A_i$, $pwasys.a_i$ and $pwasys.B_i$ would denote the dynamics of the system in region R_i.

1.3.2 Defining a PWADI model

A Piecewise-affine differential inclusion (PWADI) builds on PWA modeling. Here, instead of only one dynamics in each region R_i, we have a pair of dynamics which we may refer to them as lower- and upper-envelopes. A PWADI model, in regions R_i $(i = 1, 2, \ldots, NR)$, is described by envelopes σ_j $(j = 1, 2)$ as follows:

$$\sigma_j(x) = A_{ij}x + a_{ij} + B_{ij}u \qquad \text{if } x \in R_i$$

In fact, each σ_j is itself a PWA model. However, when we refer to PWADI we mainly mean that σ_1 and σ_2 together show a similar behavior such that they can approximate a nonlinear dynamics. In other words, if a nonlinear model is approximated by PWADIs given in (1.3.2), the nonlinear model is within the convex hull of σ_1 and σ_2. This is to say, for a nonlinear dynamics $\frac{dx}{dt}$ we will have

$$\dot{x} \in \mathbf{conv}\{\sigma_1(x), \sigma_2(x)\} \tag{1.2}$$

A PWADI model has the characteristics that if it is stabilized by a PWA controller, then any nonlinear function defined within its convex hull will be stabilized by that PWA controller [4].
Technically and similar to PWA modeling, PWADI models are stored in structures with array cells fields. Here for a PWADI model called pwainc, the fields A, a and B will be of size (NR x 2) so that, $pwainc.A_{i1}$ denotes the linear dynamics of σ_1 and $pwainc.A_{i2}$ denotes the linear dynamics of σ_2 in region R_i.

1.3.3 Region Construction In PWA /PWADI Modeling

One important issue in PWA and PWADI modeling is how the domain of definition is split into regions R_i $(i = 1, 2, \ldots, NR)$. Each region R_i is described by a convex "polyhedron" which, if bounded, it may also be called a convex "polytope" [1]. A polyhedron is bounded by several "hyperplanes" which are defined over a subset of state space. A hyperplane is described by a linear-affine equation

$$E_1 * x(1) + E_2 * x(2) + \cdots + E_n * x(n) + e = 0,$$

where $E = [E_1, E_2, \ldots, E_n]$ is a row vector, e is a scalar and one or more coefficients E_i are zero in the above definition. A polyhedron is constructed as the intersection of a finite number of half-spaces

$$E_{i1} * x(1) + E_{i2} * x(2) + \cdots + E_{in} * x(n) + e_i >= 0$$

where $E = [E_{ij}]$ is a matrix whose element at the i^{th} row and j^{th} column is E_{ij} and $e = [e_1; e_2; \ldots; e_n]$ is a column vector.
A "slab" region is a polyhedron which is described by two hyperplanes, parallel to each other. Therefore, a slab region is bounded only in one direction. We refer to region synthesis as "gridding" in our notation. Furthermore, we use the term 'Uniform' to indicate that a gridding has been done uniformly on the subset of the state space which appear in hyperplanes.
We sometime need to approximate the polytopic regions with ellipsoids. An ellipsoid equation which approximates a polytopic region is described [3] [2] [5] as

$$\varepsilon = \{x| \ \ ||Ax + b|| < 1\}$$

where A is a matrix and b is a column vector. If the regions are slab, a degenerate ellipsoid will approximate the region:

$$\varepsilon = \{x| \ \ ||Cx + d|| < 1\}$$

where C is a row vector and d is a scalar.
Furthermore, if the regions R_i and R_h $i \neq h$ share a boundary with each other, then the parametric description of the intersection is contained in

$$\bar{R}_i \cap \bar{R}_h \subseteq \{x = F_{ih}s + f_{ih} | F_{ih} \in \mathbb{R}^{n \times n-1}, \ \ f_{ih} \in \mathbb{R}^n\}$$

where $s \in \mathbb{R}^{n-1}$ is a parameter and F_{ih} is a full rank matrix.

1.4 Variable Structure and Definition In PWATOOL

Unless otherwise explicitly stated, the variables used by PWATOOL are of class "double" in MATLAB. However, there is certain structure that variables should follow in order to make them understandable by PWATOOL. In the following, we explain the structure that variables should have. However, only the types of those variables and parameters are discussed which are not of class "double" in MATLAB.

Declaring the variables in PWATOOL for creating PWA or PWADI models is easy and straightforward, based on the following rules:

1. An m-file, created by PWATOOL, will contain the models variables and parameters.
2. '*pwacreate*' command in PWATOOL creates a pre-format m-file for the user with given matrix sizes for each variable.
3. The file which is created by '*pwacreate*' can be updated, saved and run at any time to update the model.
4. The user need not take care of the name of the variables she or he needs for creating a model. All the necessary variables for creating a PWA or PWADI models will be printed in the m-file by PWATOOL in advance.
5. PWATOOL accepts the model only if the sizes of the parameters, entered by the user, are consistent with each other.

In the following, we will explain the parameters and variables which PWATOOL needs in creating a nonlinear or PWA model.

A nonlinear model in PWATOOL is of the form

$$\frac{dx}{dt} = Ax + a + f(x) + B(x)u \tag{1.3}$$

with the following parameter description:

- A : Linear dynamics
- a : Affine dynamics
- f(x): Nonlinear dynamics
- B(x): Input gain

1.4.1 Nonlinear model; Defining 'A' and 'a'

Linear (A) and affine (a) dynamics in a nonlinear model can be any variable of class "double". Examples follow:

$$A = \begin{pmatrix} 0 & 1 \\ 2 & 4 \end{pmatrix} \qquad A = rand(5) \qquad a = \begin{pmatrix} 3 \\ 5 \end{pmatrix}$$

1.4.2 Nonlinear Model; Defining $f(x)$

If the order of the system is n, then the nonlinear dynamics $f(x)$ in the nonlinear model, when *evaluated*, should generate a column vector of size (n x 1). $f(x)$ should be coded separately as a MATLAB function so that its *function handle* can be used to call and evaluate it.
Let us assume $f(x)$ is coded in a MATLAB function named 'myfunc_fx.m' as follows:

$$function\ F = \text{myfunc\_fx.m(x)}$$

$$F = \begin{bmatrix} f_1(x); \\ f_2(x); \\ \cdots \\ \cdots \\ \cdots \\ f_n(x) \end{bmatrix}$$

where $f_j(x)$ are scalar functions. Then, in the m-file which generates the nonlinear or the PWA model, the nonlinear dynamics $f(x)$ is called as follows:

Let 'NonLinearity' be the variable that contains the address of the nonlinear dynamics in the m-file. We will have

NonLinearity=@myfunc_fx;

where 'myfunc_fx.m' is the MATLAB function which we have defined previously and generates f(x) upon calling. In fact, 'NonLinearity' will contain the function handle '@myfunc_fx' and not the function. This permits the user to have the maximum degree of freedom in defining the nonlinear dynamics $f(x)$. The only restriction here is that 'myfunc_fx(x)' should be such that upon calling it a vector of size (n x 1) is generated.

1.4.3 Nonlinear model; Defining B(x)

Input gain $B(x)$ can be defined in two ways:

1. Likewise parameters 'A' and 'a', the parameter $B(x)$ can be assigned any variable of class "double";
2. Likewise nonlinear dynamics $f(x)$, the parameter $B(x)$ can be assigned a variable of class "function handle".

It should be noted that $B(x)$ should generate a matrix of size (n x m) where n is the order of the system and m is number of the inputs to the system.

1.4.4 Nonlinear Model; Defining The Domains of The Regions

The domains of the variables are defined as a cell array of size (1 x n). We write

$$Domain = \{[x_{1,min} \;\; x_{1,max}], [x_{2,min} \;\; x_{2,max}], \ldots, [x_{n,min} \;\; x_{n,max}]\}$$

where $x_{i,min}$ and $x_{i,max}$ are scalar variables. By the cell array above we mean that

- $x_{1,min}$ <= $x(1)$ <= $x_{1,max}$,
- $x_{2,min}$ <= $x(2)$ <= $x_{2,max}$,

and so on.

1.4.5 Nonlinear Model; Marking the variables which appear in $f(x)$

PWATOOL needs to know which variables appear in the nonlinear function f(x). We store the indices of these variables in a variable. We place a '1' at the j^{th} position of a vector of length 'n' for every variable $x(j)$ that appear in $f(x)$. For example, we write

$$NonlinearDomain = [0, 1, 0, 0, 1]$$

to mean that only $x(2)$ and $x(5)$ appear in $f(x)$. In other words, f(x) can be written in terms of $x(2)$ and $x(5)$ as $f(x) = g(x(2), x(5))$ where $g(.)$ is a function.

1.4.6 Nonlinear model; Gridding And The Number of The Regions

PWATOOL grids the domain of the variables which appear in $f(x)$ in three ways:

1. '**Uniform**' : The domain is gridded uniformly based on the number of regions (NR) which user chooses.
2. '**OptimalUniform**' : The domain is gridded uniformly based on the number of regions (NR) which user chooses. A norm of the approximation error is minimized in this mode.
3. '**Multiresolution**': The domain is gridded based on a predetermined given regions. In this mode, the user can set certain points in the regions a priori.

1.4.7 Nonlinear Model; Defining 'Rstar' Region

If the 'Multiresolution' is chosen as the gridding type, the user can specify a region 'Rstar' which will not be split by PWATOOL. 'Rstar' has the following applications:

- 'Rstar' can be defined such that the equilibrium point is assured to be within 'Rstar' and not on the boundaries of two or more regions.
- 'Rstar' can be defined flexibly to increase the degree of freedom in defining regions

In general 'Rstar' is a matrix of size (* x ND) where ND is the number of variables which appear in $f(x)$ and $*$ could be any number greater than or equal to 1. Each row of 'Rstar' indicates a point in the domain of $f(x)$ which will later be an edge in the final gridding. Therefore, a region

$$Rstar = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \\ x_4 & y_4 \end{bmatrix}$$

shows the points $P_1 = [x_1\ y_1]^T$, $P_2 = [x_2\ y_2]^T$, $P_3 = [x_3\ y_3]^T$ and $P_4 = [x_4\ y_4]^T$ that define a region. This ensures us that PWATOOL doesn't split the region that covers P_1 to P_4 and that the region $Rstar$ which is defined above is included in the final gridding.

1.4.8 PWA Model; Defining The Dynamics

The dynamics of a PWA model, defined by (1.1), is described by the triple $[A_i, a_i, B_i]$ which are all variables of class "double". Moreover, the domain of the variables is defined as Section 1.4.4.

1.4.9 PWA Model; Defining The Regions R_i

Each region R_i $(i = 1, 2, \ldots, NR)$ is a polytopic defined as

$$R_i = \{x|\ E_i x + e_i >= 0\},$$

where E_i is a matrix and e_i is a column vector.

Example: We write

$$E_1 = \begin{pmatrix} 1 & 0 \\ -2 & 0 \end{pmatrix} \quad \text{and} \quad e_1 = \begin{pmatrix} 0 \\ 3 \end{pmatrix}$$

to mean that $x(1) >= 0$ and $-2x(1) + 3 >= 0$ or equivalently $0 <=\ x(1)\ <= 1.5$.

1.5 Aggregated Vector Space $y = [x; 1]$

For consistency purposes, PWATOOL, also saves the system data in an aggregated vector space $y = [x; 1]$ in $\mathbb{R}^{n+1}$ where x is the variable in the vector space $\mathbb{R}^n$. The fields which hold a 'bar' in their name denote this notation. This includes 'Abar', 'Bbar', 'Ebar', 'Fbar' and 'Kbar'.

We, briefly, explain the details of this notation.

1.5.1 Aggregated Vector Space; 'Abar'

Abar contains the linear dynamics of the system. Let A_i be the linear dynamics of the PWA (or PWADI) system, a_i the affine term and 'n' the order of the system. Then, Abar is constructed as follows:

$$Abar_i = \begin{pmatrix} A_i & a_i \\ 0_{1\times n} & 1 \end{pmatrix}$$

1.5.2 Aggregated Vector Space; 'Bbar'

Bbar contains the input gain of the system. Let B_i be the input gain and 'm' the number of the inputs to the system. Then, Bbar is constructed as follows:

$$Bbar_i = \begin{pmatrix} B_i \\ 0_{1\times m} \end{pmatrix}$$

1.5.3 Aggregated Vector Space; 'Ebar'

Ebar contains the region equations matrices. Let E_i and e_i describe the region equation matrices. Then, Ebar is constructed as follows:

$$Ebar_i = \begin{pmatrix} E_i & e_i \\ 0_{1\times n} & 1 \end{pmatrix}$$

1.5.4 Aggregated Vector Space; 'Fbar'

Fbar contains the boundary equations matrices. Let F_{ij} and f_{ij} describe the boundary equation matrices. Then, Fbar is constructed as follows:

$$\begin{cases} Fbar_{ij} = [\,]; & \text{if } F_{ij} = [\,]; \\ Fbar_{ij} = \begin{pmatrix} F_{ij} & f_{ij} \\ 0_{1\times n} & 1 \end{pmatrix} & \text{if } F_{ij} \text{ is not empty} \end{cases}$$

1.5.5 Aggregated Vector Space; 'Kbar'

Kbar contains the control gains. Let K_i and k_i be the linear and affine control gain in each region. Then Kbar is constructed as follows:

$$Kbar_i = [K_i \;\; k_i] \qquad (i = 1, 2, \ldots, NR)$$

Chapter 2

Analysis of The PWA and PWADI Models; The Theory

If a PWA or PWADI model (which approximates a nonlinear model) is given, PWATOOL can analyze the model to determine if it is stable around a given equilibrium point. PWATOOL uses global- and piecewise-quadratic Lyapunov methods with both ellipsoidal and quadratic curve region approximation to analyze the system. In brief, PWATOOL analyzes a model (either PWA or PWADI) by the following four approaches:

- Global quadratic; quadratic curve approximation
- Global quadratic; ellipsoidal approximation
- PWQ; quadratic curve approximation
- PWQ; ellipsoidal approximation

For the sake of simplicity, we provide the details of above approaches for only the nonlinear models since the LMIs which are obtained for nonlinear models can easily be used for PWA models considering the number of envelopes is one (index j in Sections 2.1 and 2.2 will take only the value of l).

2.1 Global Quadratic Analysis; Nonlinear Models

Consider a PWADI model which approximates a nonlinear model (see Section 1.3.2 for details); That is for $j = 1, 2$ and in regions R_i $(i = 1, 2, \ldots, NR)$

$$\sigma_j(x) = A_{ij}x + a_{ij} + B_{ij}u \qquad \text{if } x \in R_i.$$

If the PWADI model is stable around an equilibrium point 'xcl', then the nonlinear model which is approximated by the PWADI model is also stable around xcl [4]. Note that xcl should satisfy the equilibrium point constraint, that is the dynamics should be zero at xcl.

Consider a PWADI model in which the controller gains K_i and k_i are known and the controller outputs are obtained as follows

$$u = K_i x + k_i \qquad x \in R_i$$

Note that K_i and k_i are considered zero in an open-loop system.

Now, in the global quadratic analysis, PWATOOL searches for a positive definite matrix Q that for a constant positive scalar α, make $V(x)$ a global-quadratic Lyapunov function for the PWADI system above:

$$\begin{array}{lll} (1) \quad V(x) = x^T Q x & > 0 & \text{for x defined by } \sigma_j \\ (2) \quad dV/dt = d(x^T Q x)/dt & < -\alpha V(x) & \text{for x defined by } \sigma_j \end{array}$$

If such matrix Q is found the system is stable. Now let Z_i be matrices of proper size with non-negative elements which are to be found. Then, with known controller gains K_i and k_i, Conditions (1) and (2) amount to the following LMIs for $i = 1, 2, \ldots, NR$ and $j = 1, 2$.

Notation 2.1. *In the following problem the following notations are use:*

- A_i : *linear dynamics*
- a_i : *affine term*
- B_i : *input gain*
- E_i : *region equation*
- e_i : *region equation*
- F_{ih}: *Boundary equation*
- f_{ih}: *Boundary equation*

Problem 2.2. *Global Quadratic Analysis; For known K_i and k_i and a constant positive scalar α, find Q and Z_i that satisfy the following LMIs.*

(1) $Q > 0, \;\; Z_i \succ 0$

(2) In regions R_i which contain the equilibrium point xcl

$$Q(A_{ij} + B_{ij}K_i) + (A_{ij} + B_{ij}K_i)^T Q + \alpha Q < 0$$

(3) In regions R_i which do NOT contain the equilibrium point xcl

$$\begin{bmatrix} Q(A_{ij} + B_{ij}K_i) + (A_{ij} + B_{ij}K_i)^T Q + \alpha Q + E_i^T Z_i E_i & Q(a_{ij} + B_{ij}k_i) + E_i^T Z_i e_i \\ (Q(a_{ij} + B_{ij}k_i) + E_i^T Z_i e_i)^T & e_i^T Z_i e_i \end{bmatrix} < 0$$

Remark 2.3. *PWATOOL uses Yalmip to solve above LMIs. A solution, then, may or may not be found by the solver. Since the previous LMIs are sufficient conditions for stability, we can conclude the system is stable only if we find $Q > 0$ that satisfies above LMIs. In case a matrix $Q > 0$ is not found, we may not generally be able to comment on the stability of the system.*

The regions can also be approximated by ellipsoids which in the case of slab regions they will be degenerate ellipsoids (see Section 1.3.3 for details). Ellipsoidal approximation of the regions yields to another set of LMIs.

Let ε_i be the ellipsoid that approximates a region R_i and is described as follows:

$$\varepsilon_i = \{x | ||E_i x + e_i|| < 1\}$$

Also let μ_i be the negative scalars which are to be found. Then, we should satisfy the following LMIs for $i = 1, 2, \ldots, NR$ and $j = 1, 2$

Notation 2.4. *In the following problem the following notations are use:*

- A_i : *linear dynamics*
- a_i : *affine term*
- B_i : *input gain*
- E_i : *ellipsoid parameter*
- e_i : *ellipsoid parameter*
- F_{ih}: *Boundary equation*
- f_{ih}: *Boundary equation*

Problem 2.5. *Global Quadratic Analysis; For known K_i and k_i and a constant positive scalar α, find Q and μ_i that satisfy the following LMIs.*

(1) $Q > 0, \ \ \mu_i < 0$

(2) In regions R_i which contain the equilibrium point xcl

$$Q(A_{ij} + B_{ij}K_i) + (A_{ij} + B_{ij}K_i)^T Q + \alpha Q < 0$$

(3) In regions R_i which do NOT contain the equilibrium point xcl

$$\begin{bmatrix} [Q(A_{ij} + B_{ij}K_i) + (A_{ij} + B_{ij}K_i)^T Q + \alpha Q + \mu_i E_i^T E_i & Q(a_{ij} + B_{ij}k_i) + \mu_i E_i^T e_i \\ (Q(a_{ij} + B_{ij}k_i) + \mu_i E_i^T e_i)^T & -\mu_i(1 - e_i^T e_i) \end{bmatrix} < 0$$

2.2 PWQ Analysis; Nonlinear Models

Consider a PWADI model in which the controller gains K_i and k_i are known and the controller outputs are obtained as follows

$$u = K_i x + k_i \qquad x \in R_i$$

Note that K_i and k_i are considered zero in an open-loop system.

Now, in the PWQ Lyapunov analysis, PWATOOL searches for matrices P_i, vectors q_i and scalars r_i that for a constant positive scalar α, make $V(x)$ a piecewise-quadratic Lyapunov function for the PWADI system as follows:

$$\begin{aligned} &(1) \qquad V(x) = x^T P_i x + 2q_i^T x + r_i \quad > 0 \qquad \text{for x defined by } \sigma_j \\ &(2) \qquad\qquad\qquad\qquad dV/dt \quad < -\alpha V(x) \quad \text{for x defined by } \sigma_j \end{aligned}$$

If such P_i, q_i and r_i are found, the system is stable. Let Z_i and W_i be matrices of proper sizes with non-negative elements which are to be found. Then, with known controller gains K_i and k_i, Conditions (1) and (2) amount to the following LMIs for $i = 1, 2, \ldots, NR$ and $j = 1, 2$.

Notation 2.6. *In the following problem the following notations are use:*

- A_i *: linear dynamics*
- a_i *: affine term*
- B_i *: input gain*
- E_i *: region equation*
- e_i *: region equation*
- F_{ih}*: Boundary equation*
- f_{ih}*: Boundary equation*

Problem 2.7. *PWQ Analysis; For known K_i and k_i and a constant positive scalar α, find P_i, q_i, r_i, Z_i and W_i that satisfy the following LMIs.*

(1) In regions R_i which contain the equilibrium point xcl

$$I: \quad P_i(A_{ij}+B_{ij}Ki)+(A_{ij}+B_{ij}K_i)^TP_i+\alpha P_i<0$$

$$II: \quad Z_i \succ 0,\ W_i \succ 0,\ \begin{bmatrix} P_i & q_i \\ q_i^T & r_i \end{bmatrix} > 0$$

(2) In regions R_i which do NOT contain the equilibrium point xcl

$$I: \quad \begin{bmatrix} P_i(A_{ij}+B_{ij}K_i)+(A_{ij}+B_{ij}K_i)^TP_i+\alpha P_i+E_i^TZ_iE_i & P_i(a_{ij}+B_{ij}k_i)+E_i^TZ_ie_i+(A_{ij}+B_{ij}K_i)^Tq_i \\ (P_i(a_{ij}+B_{ij}k_i)+E_i^TZ_ie_i+(A_{ij}+B_{ij}K_i)^Tq_i)^T & e_i^TZ_ie_i+2q_i^T(a_{ij}+B_{ij}k_i) \end{bmatrix} < 0$$

$$II: \quad \begin{bmatrix} P_i-E_i^TW_iE_i & -E_i^TW_ie_i+q_i \\ (-E_i^TW_ie_i+q_i)^T & -e_i^TW_ie_i+r_i \end{bmatrix} > 0$$

(3) Continuity of the Lyapunov function

$$\begin{bmatrix} F_{ih}^T(P_i-P_h)F_{ih} & F_{ih}^T(P_i-P_h)f_{ih} \\ (F_{ih}^T(P_i-P_h)f_{ih})^T & f_{ih}^T(P_i-P_h)f_{ih} \end{bmatrix} = 0 \qquad \text{for } F_{ih} \text{ not empty}$$

(4) Continuity of the control outputs

$$\begin{bmatrix} ((A_{ij}+B_{ij}K_i)-(A_{hj}+B_{hj}K_h))F_{ih} \\ ((a_{ij}+B_{ij}k_i)-(a_{hj}+B_{hj}k_h))f_{ih} \end{bmatrix} = 0 \qquad \text{for } F_{ih} \text{ not empty}$$

Again, the regions can be approximated by ellipsoid. Let ε_i be the ellipsoid that approximates a region R_i:

$$\varepsilon_i = \{x | ||E_ix+e_i|| < 1\}$$

Also let μ_i and β_i be the negative scalars which are to be found. Then, for $i = 1, 2, \ldots, NR$ and $j = 1, 2$ we should satisfy

Notation 2.8. *In the following problem the following notations are use:*

- A_i *: linear dynamics*
- a_i *: affine term*
- B_i *: input gain*
- E_i *: ellipsoid parameter*
- e_i *: ellipsoid parameter*
- F_{ih}*: Boundary equation*
- f_{ih}*: Boundary equation*

Problem 2.9. *PWQ Analysis; For known K_i and k_i and a constant positive scalar α, find P_i, q_i, r_i, μ_i and β_i that satisfy the following LMIs.*

(1) In regions R_i which contain the equilibrium point xcl

$$I: \quad P_i(A_{ij}+B_{ij}K_i)+(A_{ij}+B_{ij}K_i)^TP_i+\alpha P_i<0$$

$$II: \quad \mu_i<0,\ \ \beta_i<0,\ \ \begin{bmatrix} P_i & q_i \\ q_i^T & r_i \end{bmatrix}>0$$

(2) In regions R_i which do NOT contain the equilibrium point xcl

$$I: \quad \begin{bmatrix} P_i(A_{ij}+B_{ij}K_i)+(A_{ij}+B_{ij}K_i)^TP_i+\alpha P_i+\mu_iE_i^TE_i & P_i(a_{ij}+B_{ij}k_i)+\mu_iE_i^Te_i+(A_{ij}+B_{ij}K_i)^Tq_i \\ (P_i(a_{ij}+B_{ij}k_i)+\mu_iE_i^Te_i+(A_{ij}+B_{ij}K_i)^Tq_i)^T & -\mu_i(1-e_i^Te_i)+2q_i^T(a_{ij}+B_{ij}k_i) \end{bmatrix}<0$$

$$II: \quad \begin{bmatrix} P_i-\beta_iE_i^TE_i & -\beta_iE_i^Te_i+q_i \\ (-\beta_iE_i^Te_i+q_i)^T & \beta_i(1-e_i^Te_i)+r_i \end{bmatrix}>0$$

(3) Continuity of the Lyapunov function

$$\begin{bmatrix} F_{ih}^T(P_i-P_hF_{ih} & F_{ih}^T(P_i-P_h)f_{ih} \\ (F_{ih}^T(P_i-P_h)f_{ih})^T & f_{ih}^T(P_i-P_h)f_{ih} \end{bmatrix}=0 \qquad \text{for } F_{ih} \text{ not empty}$$

(4) Continuity of the control outputs

$$\begin{bmatrix} ((A_{ij}+B_{ij}K_i)-(A_{hj}+B_{hj}K_h))F_{ih} \\ ((a_{ij}+B_{ij}k_i)-(a_{hj}+B_{hj}k_h))f_{ih} \end{bmatrix}=0 \qquad \text{for } F_{ih} \text{ not empty}$$

2.3 Analyzing A PWA Model

A PWA model can be analyzed for stability around an equilibrium point xcl in the same way a nonlinear model is; However, for analyzing PWA models, the LMIs in Sections 2.1 and 2.2 should be solved for only $j=1$

Chapter 3

Synthesis of PWA Controllers for The PWA and PWADI Models; The Theory

PWATOOL synthesizes PWA controllers for PWA and PWADI (nonlinear) models to stabilize them around a desired equilibrium point xcl. In synthesis a PWA controller, xcl need not necessarily be an open-loop equilibrium point; It will rather be a closed-loop equilibrium point when controller gains K_i and k_i are found that stabilize the system.

PWATOOL uses the *Local Controller Extension* method [4] to design a local controller at the equilibrium point region and extend it later to the whole regions.

PWATOOL uses global and piecewise-quadratic Lyapunov methods with both ellipsoidal and quadratic curve regions approximation to synthesize PWA controllers for the model. An LMI approach, both for slab and polytopic regions, is also executed by PWATOOL. In brief, PWATOOL synthesizes PWA controllers for a model (either PWA or PWADI) by the following five methods:

- Global quadratic; quadratic curve approximation (BMI)
- Global quadratic; ellipsoidal approximation (BMI)
- PWQ; quadratic curve approximation (BMI)
- PWQ; ellipsoidal approximation (BMI)
- Global quadratic; ellipsoidal approximation (LMI)[The details of this method is not covered in this manual]

For the sake of simplicity, we only provide the details of above approaches for only the nonlinear models since the BMIs which are obtained for nonlinear models can easily be used for PWA models considering the number of envelopes is one (index j in Sections 3.1 and 3.2 will take only the value of 1).

3.1 Global Quadratic Synthesis; Nonlinear Models

Remark 3.1. *Synthesis problem for PWA or PWADI model usually yields to BMIs which in the case of global quadratic case include the LMIs which we derived in Section 2.1, excepte that, here, the controller gains K_i and k_i are unknown changing the previous LIMs into BMIs. In addition, we optionally add a constraint, called the* continuity of the control inputs*, to other constraints so that the controllers yield a smoother behavior.*

Consider a PWADI model which approximates a nonlinear model (see Section 1.3.2 for details); That is for $j = 1, 2$ and in regions R_i $(i = 1, 2, \ldots, NR)$

$$\sigma_j(x) = A_{ij}x + a_{ij} + B_{ij}u \qquad \text{if } x \in R_i.$$

PWATOOL stabilizes the PWADI model around a desired equilibrium point xcl and hence the PWA controller which is obtained will stabilize the nonlinear model around that desired equilibrium point xcl. Consider a PWADI model in which the controller outputs is obtained as follows

$$u = K_i x + k_i.$$

Now, in the global quadratic synthesis problem, PWATOOL searches for a positive definite matrix Q that for a constant positive scalar α, make $V(x)$ a global quadratic Lyapunov function for the PWADI system above:

$$\begin{array}{lll} (1)\ \ V(x) = x^T Q x & > 0 & \text{for x defined by } \sigma_j \\ (2)\ \ dV/dt = d(x^T Q x)/dt & < -\alpha V(x) & \text{for x defined by } \sigma_j \end{array}$$

Conditions (1) and (2) amount to the BMIs in Problem 3.3:

Notation 3.2. *In the following problem the following notations are use:*

- A_i *: linear dynamics*
- a_i *: affine term*
- B_i *: input gain*
- E_i *: region equation*
- e_i *: region equation*
- F_{ih}*: Boundary equation*
- f_{ih}*: Boundary equation*

Problem 3.3. *Global Quadratic Synthesis; For a constant positive scalar α, find Q, K_i, k_i and Z_i that satisfy the following BMIs for $i, h = 1, 2, \ldots, NR$ and $j = 1, 2$:*

(1) $Q > 0, \ \ Z_i \succ 0$

(2) In regions R_i which contain the equilibrium point xcl

$$Q(A_{ij} + B_{ij}K_i) + (A_{ij} + B_{ij}K_i)^T Q + \alpha Q < 0$$

(3) In regions R_i which do NOT contain the equilibrium point xcl

$$\left[\begin{array}{ll} Q(A_{ij} + B_{ij}K_i) + (A_{ij} + B_{ij}K_i)^T Q + \alpha Q + E_i^T Z_i E_i & Q(a_{ij} + B_{ij}k_i) + E_i^T Z_i e_i \\ (Q(a_{ij} + B_{ij}k_i) + E_i^T Z_i e_i)^T & e_i^T Z_i e_i \end{array} \right] < 0$$

(4) Continuity of the control outputs (OPTIONAL)

$$\left[\ ((A_i + B_i K_i) - (A_h + B_h K_h)) * F_{ih}((a_i + B_i k_i) - (a_h + B_h k_h)) * f_{ih} \ \right] = 0 \qquad \text{for } F_{ih} \text{ not empty}$$

The regions can also be approximated with ellipsoids which in the case of slab regions they will be degenerate ellipsoids (see Section 1.3.3 for details). Ellipsoidal approximation of the regions yields to another set of BMIs:

Let ε_i be the ellipsoid that approximates a region R_i and is described as follows:

$$\varepsilon_i = \{x | ||E_i x + e_i|| < 1\}$$

Then, the BMIs in Problem 3.5 are obtained.

Notation 3.4. *In the following problem the following notations are use:*

- A_i *: linear dynamics*
- a_i *: affine term*
- B_i *: input gain*
- E_i *: ellipsoid parameter*
- e_i *: ellipsoid parameter*
- F_{ih}*: Boundary equation*
- f_{ih}*: Boundary equation*

Problem 3.5. *Global Quadratic Synthesis; For a constant positive scalar α, find Q, K_i, k_i and μ_i that satisfy the following BMIs for $i, h = 1, 2, \ldots, NR$ and $j = 1, 2$:*

(1) $Q > 0, \ \ \mu_i < 0$

(2) In regions R_i which contain the equilibrium point xcl

$$Q(A_{ij} + B_{ij}K_i) + (A_{ij} + B_{ij}K_i)^T Q + \alpha Q < 0$$

(3) In regions R_i which do NOT contain the equilibrium point xcl

$$\begin{bmatrix} [Q(A_{ij} + B_{ij}K_i) + (A_{ij} + B_{ij}K_i)^T Q + \alpha Q + \mu_i E_i^T E_i & Q(a_{ij} + B_{ij}k_i) + \mu_i E_i^T e_i \\ (Q(a_{ij} + B_{ij}k_i) + \mu_i E_i^T e_i)^T & -\mu_i(1 - e_i^T e_i) \end{bmatrix} < 0$$

(4) Continuity of the control outputs (OPTIONAL)

$$\begin{bmatrix} ((A_i + B_iK_i) - (A_h + B_hK_h)) * F_{ih} \\ ((a_i + B_ik_i) - (a_h + B_hk_h)) * f_{ih} \end{bmatrix} = 0 \qquad \text{for } F_{ih} \text{ not empty}$$

3.2 PWQ Synthesis; Nonlinear Models

Remark 3.6. *The BMIs in synthesis by PWQ Lyapunov approach are in fact the LMIs which we derived for analysis in Section* **??***, except that, here, the controller gains K_i and k_i are unknown changing the previous LIMs into BMIs.*

Consider a PWADI model in which the controller outputs is obtained as follows

$$u = K_i x + k_i.$$

Now, in the PWQ Lyapunov synthesis approach, PWATOOL searches for matrices P_i, vectors q_i and scalars r_i that for a constant positive scalar α, make $V(x)$ a PWQ Lyapunov function for the PWADI system above:

$$\begin{array}{lll} (1) \quad V(x) = x^T P_i x + 2q_i^T x + r_i & > 0 & \text{for x defined by } \sigma_j \\ (2) \quad dV/dt & < -\alpha V(x) & \text{for x defined by } \sigma_j \end{array}$$

Then, conditions (1) and (2) yield the BMIs in Problem 3.8.

Notation 3.7. *In the following problem the following notations are use:*

- A_i *: linear dynamics*
- a_i *: affine term*
- B_i *: input gain*
- E_i *: region equation*
- e_i *: region equation*
- F_{ih}*: Boundary equation*
- f_{ih}*: Boundary equation*

Problem 3.8. *PWQ Synthesis; For a constant positive scalar α, find P_i, q_i, r_i, K_i, k_i, Z_i and W_i that satisfy the following BMIs for $i, h = 1, 2, \ldots, NR$ and $j = 1, 2$:*

(1) In regions R_i which contain the equilibrium point xcl

$$I: \quad P_i(A_{ij}+B_{ij}Ki)+(A_{ij}+B_{ij}K_i)^TP_i+\alpha P_i<0$$

$$II: \quad Z_i \succ 0,\ W_i \succ 0,\ \begin{bmatrix} P_i & q_i \\ q_i^T & r_i \end{bmatrix} > 0$$

(2) In regions R_i which do NOT contain the equilibrium point xcl

$$I: \quad \begin{bmatrix} P_i(A_{ij}+B_{ij}K_i)+(A_{ij}+B_{ij}K_i)^TP_i+\alpha P_i+E_i^TZ_iE_i & P_i(a_{ij}+B_{ij}k_i)+E_i^TZ_ie_i+(A_{ij}+B_{ij}K_i)^Tq_i \\ (P_i(a_{ij}+B_{ij}k_i)+E_i^TZ_ie_i+(A_{ij}+B_{ij}K_i)^Tq_i)^T & e_i^TZ_ie_i+2q_i^T(a_{ij}+B_{ij}k_i) \end{bmatrix} < 0$$

$$II: \quad \begin{bmatrix} P_i-E_i^TW_iE_i & -E_i^TW_ie_i+q_i \\ (-E_i^TW_ie_i+q_i)^T & -e_i^TW_ie_i+r_i \end{bmatrix} > 0$$

(3) Continuity of the Lyapunov function

$$\begin{bmatrix} F_{ih}^T(P_i-P_h)F_{ih} & F_{ih}^T(P_i-P_h)f_{ih} \\ (F_{ih}^T(P_i-P_h)f_{ih})^T & f_{ih}^T(P_i-P_h)f_{ih} \end{bmatrix} = 0 \qquad \text{for } F_{ih} \text{ not empty}$$

(4) Continuity of the control outputs

$$\begin{bmatrix} ((A_{ij}+B_{ij}K_i)-(A_{hj}+B_{hj}K_h))F_{ih} \\ ((a_{ij}+B_{ij}k_i)-(a_{hj}+B_{hj}k_h))f_{ih} \end{bmatrix} = 0 \qquad \text{for } F_{ih} \text{ not empty}$$

Again, the regions can be approximated by ellipsoids. Let ε_i be the ellipsoid that approximates a region R_i:

$$\varepsilon_i = \{x | ||E_ix+e_i|| < 1\}$$

Then, the BMIs in Problem 3.10 are obtained.

Notation 3.9. *In the following problem the following notations are use:*

- A_i *: linear dynamics*
- a_i *: affine term*
- B_i *: input gain*
- E_i *: ellipsoid parameter*
- e_i *: ellipsoid parameter*
- F_{ih}*: Boundary equation*
- f_{ih}*: Boundary equation*

Problem 3.10. *PWQ Synthesi; For a constant positive scalar α, find P_i, q_i, r_i, K_i, k_i, μ_i and β_i that satisfy the following BMIs for $i = 1, 2, \ldots, NR$ and $j = 1, 2$:*

(1) In regions R_i which contain the equilibrium point xcl

$$I: \quad P_i(A_{ij}+B_{ij}K_i)+(A_{ij}+B_{ij}K_i)^TP_i+\alpha P_i<0$$

$$II: \quad \mu_i<0,\ \ \beta_i<0,\ \ \begin{bmatrix} P_i & q_i \\ q_i^T & r_i \end{bmatrix}>0$$

(2) In regions R_i which do NOT contain the equilibrium point xcl

$$I: \quad \begin{bmatrix} P_i(A_{ij}+B_{ij}K_i)+(A_{ij}+B_{ij}K_i)^TP_i+\alpha P_i+\mu_iE_i^TE_i & P_i(a_{ij}+B_{ij}k_i)+\mu_iE_i^Te_i+(A_{ij}+B_{ij}K_i)^Tq_i \\ (P_i(a_{ij}+B_{ij}k_i)+\mu_iE_i^Te_i+(A_{ij}+B_{ij}K_i)^Tq_i)^T & -\mu_i(1-e_i^Te_i)+2q_i^T(a_{ij}+B_{ij}k_i) \end{bmatrix}<0$$

$$II: \quad \begin{bmatrix} P_i-\beta_iE_i^TE_i & -\beta_iE_i^Te_i+q_i \\ (-\beta_iE_i^Te_i+q_i)^T & \beta_i(1-e_i^Te_i)+r_i \end{bmatrix}>0$$

(3) Continuity of the Lyapunov function

$$\begin{bmatrix} F_{ih}^T(P_i-P_hF_{ih} & F_{ih}^T(P_i-P_h)f_{ih} \\ (F_{ih}^T(P_i-P_h)f_{ih})^T & f_{ih}^T(P_i-P_h)f_{ih} \end{bmatrix}=0 \qquad \text{for } F_{ih} \text{ not empty}$$

(4) Continuity of the control outputs

$$\begin{bmatrix} ((A_{ij}+B_{ij}K_i)-(A_{hj}+B_{hj}K_h))F_{ih} \\ ((a_{ij}+B_{ij}k_i)-(a_{hj}+B_{hj}k_h))f_{ih} \end{bmatrix}=0 \qquad \text{for } F_{ih} \text{ not empty}$$

3.3 Synthesis PWA controllers for A PWA Model

PWATOOL synthesize PWA controllers for a PWA model in the same way it does for a nonlinear models is; However, for synthesizing PWA controllers for the PWA models, the BMIs in Sections 3.1 and 3.2 should be solved for only $j=1$

Chapter 4

Examples; Analysis and Synthesis

In this chapter we give several examples to show how PWATOOL, in practice, works. This includes creating a model (PWA / PWADI), analyzing the model and synthesizing PWA controllers for it. Not all the design approaches that were given for the analysis and synthesis converge for every example. Therefore, in each example, some of the approaches for analysis or synthesis may not converge.

4.1 Nonlinear Resistor Example; Modeling, Analysis and Synthesis

We use the nonlinear resistor example [3] to show how PWATOOL can be used for modeling, analysis and synthesis. Figure 4.1 shows an RLC circuit where the current through and the voltages across the resistor have a nonlinear relation with respect to each other. Figure 4.2 shows this nonlinear relation. The KVL equations governing the electrical circuit in Figure 4.1 is

$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{pmatrix} -30 & -20 \\ .05 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} 24 \\ -50g(x_2) \end{pmatrix} + \begin{pmatrix} 20 \\ 0 \end{pmatrix} u$$

where the nonlinear function $g(x_2)$ is characterized in Figure 4.2. We would like to design a PWA controller to stabilize the system around the open-loop equilibrium operating point xcl as given below:

$$xcl = \begin{bmatrix} 0.371428571428570 \\ \\ 0.642857142857146 \end{bmatrix}$$

A road map of the design steps follows.

1. Approximation of the nonlinear model with PWADIs (modeling)
2. Checking the stability of the open-loop system around xcl (analysis)
3. Designing PWA controllers for stabilizing the system around xcl (synthesis)
4. Checking the stability of the closed-loop system around xcl (analysis)

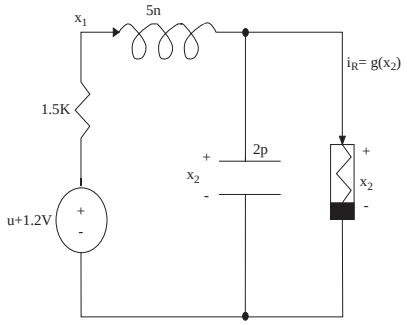

Figure 4.1: Circuit with nonlinear resistor [3]

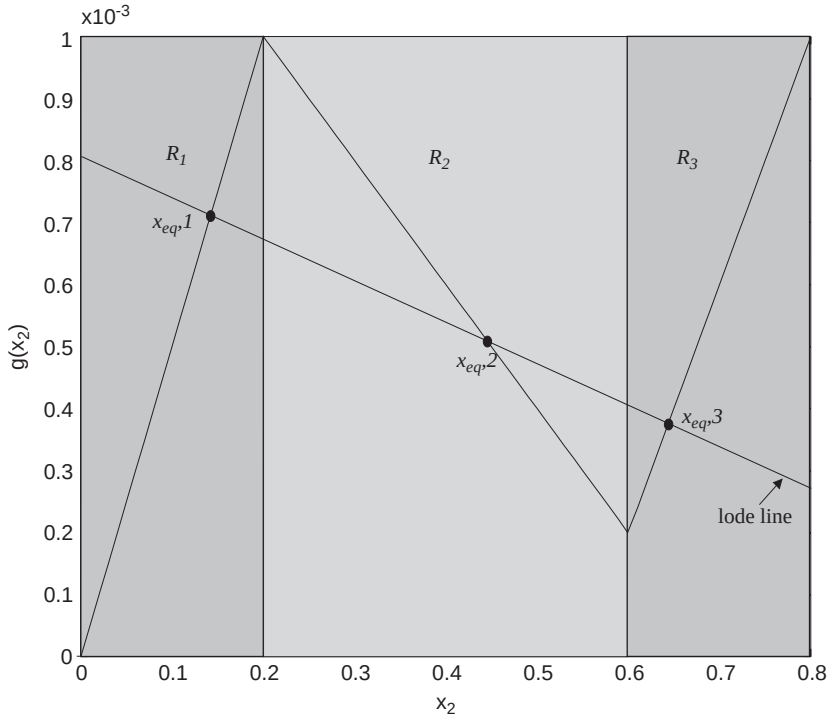

Figure 4.2: Nonlinear resistor characteristics [3]

4.1.1 Creating a nonlinear model

By "Creating a nonlinear model" we mean to save the nonlinear dynamics that user enters and approximate it with PWADIs. Figure 4.3 gives a general overview of how PWATOOL approximates a nonlinear model with PWADIs.

The diagram in left side of Figure 4.3 shows the modeling steps from the user's perspective while the tables in the right side show the internal components of the two important functions which check the data (*NonCheckModel*) and generate the approximations (*PWAComp*).

CREATING AN M-FILE: As the first step, we use the command *pwacreate* to build an m-file for entering data. The system is of the order two (n=2) and has one input (m=1). Therefore, we type

pwacreate(2, 1, 'nonlinear_resistor_non.m')

By doing so, an m-file called 'nonlinear_resistor_non.m' will be created (if does not exist before) or rewritten (if existed before) in the current directory.

Now open 'nonlinear_resistor_non.m'. This file will contain the complete dynamics of the systems. Note that the file has two sections of which, only Section 1 should be filled by the user. Section 2 should remain as is to let the code be executed after completion.

The parameters below have been generated from the data that user has passed to 'pwacreate' and should not be changed by the user.

```
model.type='nonlinear';
model.dim=[2, 1];
```

LINEAR, AFFINE DYNAMICS AND INPUT GAIN: PWATOOL accepts the dynamics of a nonlinear model based on the parameter description given in (1.3). The Linear and affine dynamics of the nonlinear system are matrices of sizes (n x n) and (n x 1). We assume a variable called 'model' is to contain the system data. We will have

```
model.A=[-30   -20              model.aff=[24
         .05    0]                         0];
```

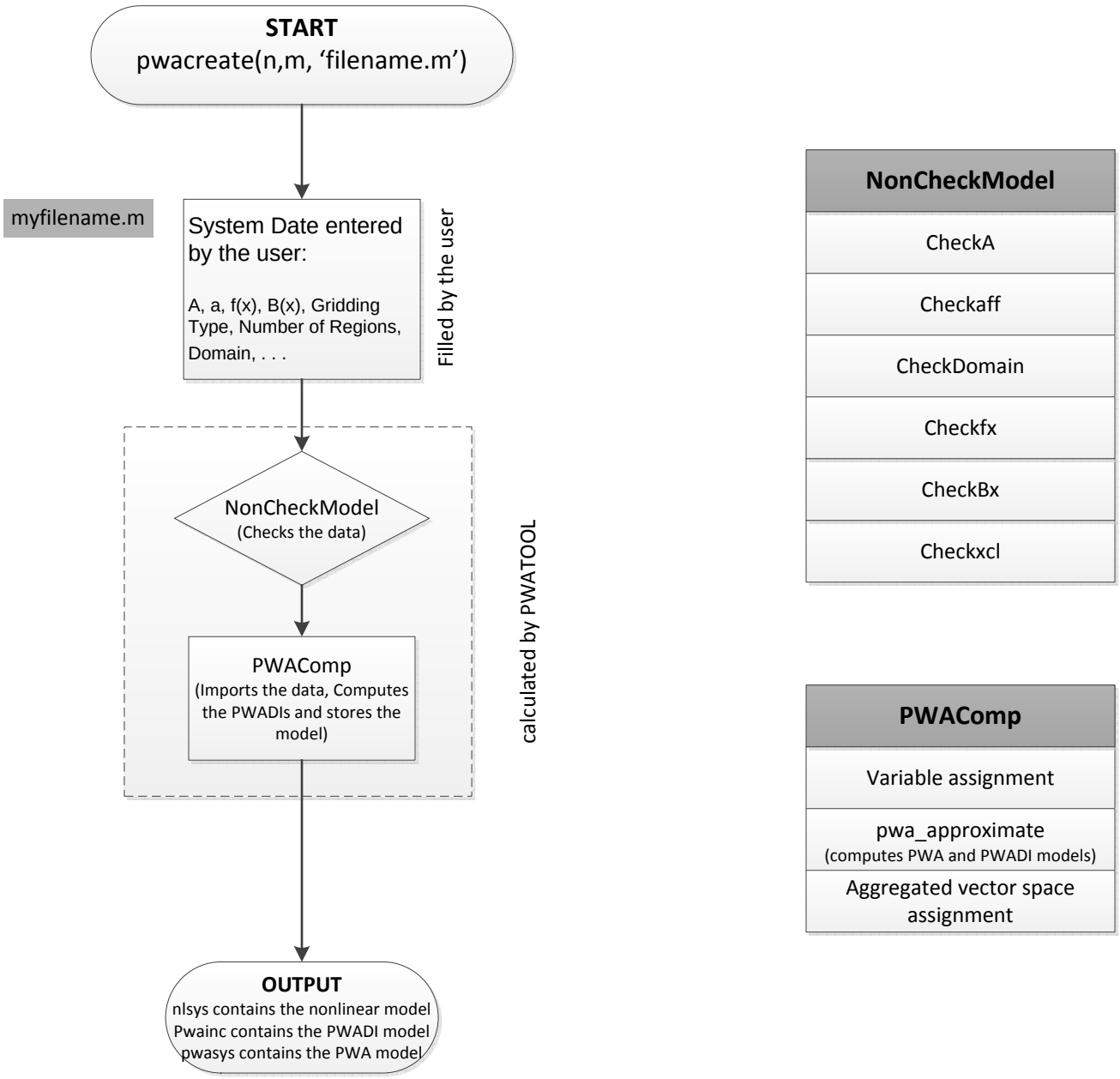

Figure 4.3: Creation of a nonlinear model

In this example, the input gain is a constant matrix of size (n x m). We have

```
model.Bx=[20
          0];
```

NONLINEARITY: The nonlinear dynamics $f(x)$ is coded separately in another m-file as a **function**. Then, in order to use that function, the function handle of the m-file which contains the function is used in 'nonlinear_resistor_non.m'.

Here, we have coded the nonlinear function in an m-file called 'resistor_nonlinearity.m'. Therefore, we will write

```
model.fx=@resistor_nonlinearity;
```

where the m-file 'resistor_nonlinearity.m' has been coded as follows:

```
 ---------------------------------------
| function F = resistor_nonlinearity(X) |
| x=X(2);                               |
| if (-2e4 <= x & x< .2)                |
|     F=[0; -50*(1e-3/.2*x)];           |
| elseif (.2 <= x & x<.6)               |
|     F=[0; -50*(-2e-3*x+1.4e-3)];      |
| elseif (.6 <= x & x <= 2e4)           |
|     F=[0; -50*(4e-3*x-2.2e-3)];       |
| end                                   |
 ---------------------------------------
```

Remark 4.1. *The output of the function generated above should be a column vector of size (n x 1). Hence, some elements of the vector 'F' above are set to zero since they are not set by the function.*

DOMAIN: The domain of the variables is defined as

```
model.Domain={[-20  20],[-2e4   2e4]};
```

There is also one more variable which we call it 'NonlinearDomain'. This is, in fact, the indices of the variables which appear in the nonlinear function. 'NonlinearDomain' is a row vector of size (1 x n) in which for every variable x(j) which appear in the nonlinear dynamics $f(x)$ we put a '1' at the j^{th} place of the vector model.NonlinearDomain. We will have

```
model.NonlinearDomain=[0  1];
```

to mean that model.fx is a function of only $x(2)$.

EQUILIBRIUM POINT: model.xcl is a column vector of size (n x 1) and shows the "desired" equilibrium point:

```
model.xcl=[0.371428571428570
           0.642857142857146];
```

NUMBER OF REGIONS AND GRIDDING TYPE: model.NR is a vector of size (1 x ND) where ND is the number of the variables which appear in model.fx. Elements of model.NR show the number of the regions which we like to have at each direction corresponding to x(j) which appear in model.fx. If model.NR is given as a scalar, all of the variables x(j) which appear in model.fx, will have the same number of regions in their corresponding direction. For the nonlinear resistor example we have

```
model.NR=3;
```

The domains of the variables which appear in model.fx is gridded by PWATOOL. The current types of gridding are 'Uniform', 'OptimalUniform', 'Multiresolution'. In our example we will have

```
model.mtd='Multiresolution';
```

RSTAR REGION: We choose the Rstar region based on the characteristics we have in Figure 4.2 (see Chapter 1 for the definition of Rstar)

```
model.Rstar=[.2 ; .6];
```

By specifying above Rstar and since the total number of the regions is 3, the regions which we obtain will be $R_1 = [-2e4 \ \ .2]$, $R_2 = [.2 \ \ .6]$ and $R_3 = [.6 \ \ 2e4]$.

CHECKING THE DATA: Now, save the changes in the file 'nonlinear_resistor_non.m' and run it by either pressing F5 or typing 'run nonlinear_resistor_non'. PWATOOL imports the data that user has entered and checks for dimension consistency between the model parameters by calling the function

```
[Err, model]=NonCheckModel(model);
```

BUILDING THE PWADI APPROXIMATION: If there is no error ($Err = 0$), function *PWAComp* is called to build the PWADI approximation.

```
[pwainc, pwasys, nlsys]=PWAComp(model);
```

The outputs that PWAComp generates are as follows:

- nlsys : The nonlinear model data that user has enterted
- pwainc: PWADI approximation of the nonlinear model
- pwasys: PWA approximation of the nonlinear model

This is a summary of the fields of pwainc which contains the PWADI approximation of the system:

PARAMETER	STRUCTURE	REPRESENTRS	Size of Each Matrix / Vector
A	NR x 2 cell	Linear dynamics	n x n
a	NR x 2 cell	affine term	n x 1
B	NR x 2 cell	Input gain	n x m
E	NR x 1 cell	Regions equation	* x n
e	NR x 1 cell	Regions equations	* x 1
F	NR x NR cell	Boundary equations	n x (n-1)
f	NR x NR cell	Boundary equations	n x 1
Abar	NR x 2 cell	Linear dynamics	(n+1) x (n+1)
Bbar	NR x 2 cell	Input gain	(n+1) x m
Ebar	NR x 1 cell	Regions equation	* x (n+1)
Fbar	NR x NR cell	Boundary equations	(n+1) x (n+1)
NR	1 x ** array	Number of regions	1 x **

$*$: The number of rows in matrix E and vector e depends on the gridding type. If the system is slab, then $[2, n] = size(E)$; if the gridding type is uniform and the system is not slab, then $[n + 1, n] = size(E)$ and $[n + 1, 1] = size(e)$.

4.1.2 Stability Analysis Of The Open-loop System

To check the stability of PWADIs (nonlinear model) around an equilibrium point we use the *pwaanalysis* command. Other than the system model and the equilibrium point, *pwaanalysis* has several optional controls which provide user with more flexibility in analyzing the model. The user can restrict the type of Lyapunov function (PWQ or global) and the type of the approximation for the regions (quadratic curve or ellipsoidal) for analysis. The *pwaanalysis* command is used as

```
pwaanalysis(pwainc, setting);
```

where 'setting' is a structured variable.

ANALYSIS PARAMETERS: We explain the different controls and parameters in 'setting'.

setting.Lyapunov: shows whether 'global' or 'pwq' Lyapunov function should be used in analysis

setting.ApxMeth: shows whether 'ellipsoidal' or 'quadratic curve' method should be used to approximate the regions.

setting.alpha: specifies the decay rate in finding the Lyapunov function Generally, bigger values of alpha usually make the convergence harder

setting.xcl: shows the desired equilibrium point which user sets. If it is not set by the user, PWATOOL uses the pwainc.xcl (if exists) as the equilibrium point. PWATOOL issues an error and stops running if xcl is not defined or if at xcl we cannot satisfy the equilibrium point equations analytically.

For the nonlinear resistor model we set

```
setting.Lyapunov='pwq';

setting.alpha=.1;

setting.xcl=[0.371428571428570;
             0.642857142857146];
```

By doing so, *setting.ApxMeth* will take the largest possible set by default. In other words $setting.ApxMeth = \{'quadratic', 'ellipsoidal'\}$.

ANALYSIS AND RESULTS: We can now analyze the nonlinear system

```
pwaanalysis(pwainc, setting);
```

By the above command, PWATOOL analyzes the PWADIs (nonlinear system) by two approaches:

- PWQ; quadratic curve approximation
- PWQ; ellipsoidal approximation

It, then, prints a message that

```
I could not verify if the open-loop nonlinear system is stable at xcl.
```

It implies that PWATOOL has not been able to solves the LMIs. Therefore, the open-loop equilibrium point xcl may or may not be a stable equilibrium point for the system.

4.1.3 Synthesis Of PWA Controllers

PWA controllers can be synthesized by PWATOOL by the following command:

```
ctrl = pwasynth(pwainc, x0, setting);
```

where 'pwainc' is the PWADI approximation of the nonlinear model, x0 is the initial point for simulation and 'setting' a structured variable. We start with explaining the fields of setting. Note that most of the fields of 'setting', if not set by the user, will take default values by PWATOOL.

PARAMETERS OF SYNTHESIS:

setting.Lyapunov: shows whether 'global' or 'pwq' Lyapunov function should be used in the synthesis

setting.ApxMeth: shows whetehr 'ellipsoidal' or 'quadratic' curve method should be used to approximate the regions.

setting.SynthMeth: shows whether 'LMI' or 'BMI' methods should be used.

setting.QLin: positive definite matrix of size n x n where n is the order of the system. It is used as the LQR parameter for the region which contains the equilibrium point

setting.RLin: positive definite matrix of size m x m where m is the number of the inputs to the system. It is used as the LQR parameter for the region which contains the equilibrium point

setting.RandomQ: if set, shows setting.QLin is set randomly by PWATOOL. If it is zero, PWATOOL keeps the value that user enters for setting.QLin

setting.RandomR: if set, shows setting.RLin is set randomly by PWATOOL. If it is zero, PWATOOL keeps the value that user enters for setting.RLin

setting.alpha: specifies the decay rate of the closed-loop system. Generally, the bigger alpha is, the faster the dynamics of the system will be. However, bigger values of alpha make the convergence of the controller design harder.

setting.NormalDirectionOnly: specifies if only the normal directions at the boundaries should meet the continuity constraints. By default, it would be zero so that the whole directions are included in the continuity constraints. In that case, if convergence is obtained, the results are usually smoother than those obtained by NormalDirectionOnly set to 1.

setting.xcl: shows the desired equilibrium point which user sets. If it is not set by the user, PWATOOL uses the pwainc.xcl (if exists) as the equilibrium point. PWATOOL issues an error and stops running if it cannot satisfy the equilibrium point equations analytically at xcl.

setting.StopTime: shows the simulation time in seconds. If not set by the user, it will be set to 10 sec.

setting.IterationNumber: shows the maximum number of times that PWATOOL tries to synthesize PWA controller if it fails in doing so in the first try. The default value is 5.

For the nonlinear resistor example we set

```
x0=[1.2 .23]';

setting.SynthMeth='bmi';

setting.QLin = [127.9038   55.4185
                55.4185   37.7115];

setting.RLin =  30.0877;

setting.RandomQ=0;

setting.RandomR=0;

setting.alpha=.1;

setting.NormalDirectionOnly=0;
```

SYNTHESIS AND RESULTS: Based on the settings that we have chosen for the fields 'Lyapunov', 'ApxMeth' and 'SynthMeth', PWATOOL tries synthesizing PWA controllers by the following methods (LMI approach has been removed from the possibilities due to our selection for settings)

1. BMI approach with ellipsoidal approximation (golabl Lyapunov)
2. BMI approach with quadratic curve approximation (golabl Lyapunov)
3. BMI approach with ellipsoidal approximation (pwq Lyapunov)
4. BMI approach with quadratic curve approximation (pwq Lyapunov)

We type

```
ctrl = pwasynth(pwainc, x0, setting);
```

Then, PWATOOL starts synthesizing PWA controllers by above methods. The following message is printed after all of the methods above are run by PWATOOL:

```
PWATOOL was successful in synthesizing PWA controllers by the following method(s)

BMI approach with QUADRATIC CURVE approximation (GLOBAL Lyapunov)
```

In the case of a successful design a Simulink model is also called to simulate the closed-loop system with either of the achieved stabilizing PWA controllers. Furthermore, a variable called *pwacont* will contain the PWA controllers and the corresponding methods of convergence. *pwacont* is essentially equal to the variable *ctrl* which we may define as the output in calling *pwasynth*.

```
pwacont =

            Gain: {{1x3 cell}}
       AppMethod: {'QUADRATIC CURVE'}
   SynthesisType: {'BMI'}
        Lyapunov: {'Global'}
     Sprocedure1: {{1x3 cell}}
     Sprocedure2: {{1x3 cell}}
```

pwacont.Gian: contains the controller which has been designed (Kbar), in the aggregated vector space. For example *pwacont.Gain*{1} will be a cell of size (1 x n) where each element would contain the controller $Kbar_i = [K_i \;\; k_i]$ for region R_i $(i = 1, 2, \ldots, NR)$.

```
>> pwacont.Gain{1}{:}

ans =

   -1.0500 -394.8175   26.6046


ans =

   -1.0500   -0.3326         0


ans =

   -1.0500 -503.5952  -21.5684
```

Remark 4.2. *If none of the methods, among the possibilities, converge in the first try, PWATOOL repeats the synthesis to a maximum of 'IterationNumber' times which by default is set to 5. In that case, PWATOOL sets the fields* RandomQ *and* RandomQ *to 1.*

4.1.4 Stability Analysis Of The Closed-loop System

PWATOOL can check the stability of a closed-loop system too. To this regard, the *pwaanalysis* function may be called by three inputs where the third input to the functions is a PWA controller. In our example, we can test this capability by typing

```
pwaanalysis(pwainc, setting, pwacont.Gain{1});
```

Here, *pwacont.Gain*{*1*} is the PWA controller gains which we obtained in previous section in the aggregated vector space $y = [x; 1]$.

PWATOOL will check the PWADIs, contained in pwainc, for stability, this time with the assumption that the loop is closed through the PWA controllers *pwacont.Gain*{*1*}. PWATOOL prints the following message after analysis:

```
The following methods verified that the closed-loop nonlinear system is stable at xcl.

ELLIPSOIDAL     approximation (Global Lyapunov function)
QUADRATIC CURVE approximation (Global Lyapunov function)
```

This concludes the nonlinear resistor example.

Bibliography

[1] ttp://www.wordiq.com/definition/polyhedron. April 2011.

[2] A. Hassibi and S. Boyd. Quadratic stabilization and control of piecewise-linear systems. *Proceedings of the American Control Conference*, 6:3659 – 3664, 1998.

[3] L. Rodrigues and S. Boyd. Piecewise-affine state feedback for piecewise-affine slab systems using convex optimization. *Systems and Control Letters*, 54:835–853, 2005.

[4] B. Samadi and L. Rodrigues. Extension of local linear controllers to global piecewise affine controllers for uncertain non-linear systems. *International Journal of Systems Science*, 39(9):867–879, 2008.

[5] L. Vandenberghe, S. Boyd, and S.P. Wu. Determinant maximization with linear matrix inequality constraints. *SIAM Journal on Matrix Analysis and Applications*, 9(2):499–533, 1998.