Technische Universität Chemnitz

Sonderforschungsbereich 393

Numerische Simulation auf massiv parallelen Rechnern

M. Randrianarivony G. Brunnett

Parallel implementation of curve reconstruction from noisy samples

Preprint SFB393/02-15

Abstract

This paper is concerned with approximating noisy samples by non-uniform rational B-spline curves with special emphasis on free knots. We show how to set up the problem such that nonlinear optimization methods can be applied efficiently. This involves the introduction of penalizing terms in order to avoid undesired knot positions. We report on our implementation of the nonlinear optimization and we show a way to implement the program in parallel. Parallel performance results are described. Our experiments show that our program has a linear speedup and an efficiency value close to unity. Runtime results on a parallel computer are displayed.

Keywords: rational B-splines, curve reconstruction, parallel implementation, noisy samples.

Preprint-Reihe des Chemnitzer SFB 393

ISSN 1619-7178 (Print) **ISSN 1619-7186** (Internet)

SFB393/02-15 **August 2002**

Contents

Author's addresses:

M. Randrianarivony
TU Chemnitz
Fakultät für Informatik
D-09107 Chemnitz
maharavo@informatik.tu-chemnitz.de
`http://www.tu-chemnitz.de/~ranh`

G. Brunnett
TU Chemnitz
Fakultät für Informatik
D-09107 Chemnitz
brunnett@informatik.tu-chemnitz.de
`http://www.informatik.tu-chemnitz.de/~gdv`

1 Introduction

Non-uniform rational B-splines (NURBS) settings are appreciated in many applications because they allow flexible description of both free form surfaces and usual geometries such as conic sections. Indeed, the set of rational functions is much larger than that of polynomial functions, so NURBS give mainly better approximations than their polynomial B-spline counterparts do. Another reason for the appreciation of NURBS is that they are supported by many softwares. For instance, it is easy in OpenGL and ACIS ([11], [2]) to create and manipulate NURBS by just giving the required parameters.

Our interest in the subject of approximation with NURBS is motivated by the application of reverse engineering. In reverse engineering, one is concerned with the automated generation of a computer aided design model from a set of points digitized from an existing 3D object. Since many real world objects have been constructed using both simple algebraic surfaces as well as free-form surfaces, NURBS surfaces appear to be a universal class for surface fitting in reverse engineering.

In this paper we consider the curve case as a preliminary investigation for the surface case. We assume that a sequence of noisy sample points $\mathbf{M}_i$ is given and we aim at reconstructing a NURBS curve $\mathbf{X}$ that approximates the points in a least-square sense.

It is well known (see [7]) that the choice of the knot vector of a spline (also called parameterization) has a tremendous influence on the result of the fitting procedure. For this reason, several suggestions for a reasonable knot spacing have been made (see works of Foley, Nielson, Lee which are referenced in [7]). However, for all these suggestions of knot spacings, examples can be found where these methods provide unsatisfactory results. Furthermore, these methods can only be applied for interpolatory splines while we are interested in spline approximation. Therefore, for reliable results, one has to treat the knots as unknowns in the approximation process.

In the context of polynomial B-splines, the use of free knots has already been investigated by several authors ([18, 9, 17]). For the case of NURBS, the following approaches have been taken: in [4], the author uses an iterative segment determination in order to establish the positions of the knots. The authors of [10] use an improved version of Polak-Ribiere algorithm in order to minimize some cost functional without trying to reduce the number of parameters.

In this paper, we report on our implementation of a general method for approximation by NURBS curves with free knots. In section 2, we provide the necessary preliminaries to state the approximation problem. In section 3, we reformulate the problem such that nonlinear optimization methods can be applied efficiently. For this, we reduce the dimensionality of the minimization problem and introduce penalizing terms in order to avoid undesired knot positions. In section 4, we describe in detail our implementation of the nonlinear optimization that is based on the Levenberg-Marquardt method. In the last section, we discuss the results obtained by this approach for several data sets and different knot spacings.

2 Problem setting and notations

2.1 Open and closed NURBS curves

A nonuniform rational B-spline curve (NURBS curve) with weights $w_0, ..., w_n \in \mathbf{R}_+$ and control points $\mathbf{d}_0, ..., \mathbf{d}_n \in \mathbf{R}^3$ is given by:

$$\mathbf{X}(t) := \frac{\sum_{i=0}^{n} w_i \mathbf{d}_i N_i^k(t)}{\sum_{i=0}^{n} w_i N_i^k(t)}, \tag{1}$$

where N_i^k is the usual B-spline basis ([5]) defined recursively by:

$$N_i^1(t) := \begin{cases} 1 & \text{if } t \in [\theta_i, \theta_{i+1}) \\ 0 & \text{otherwise} \end{cases}$$

$$N_i^k(t) = \frac{t - \theta_i}{\theta_{i+k-1} - \theta_i} N_i^{k-1}(t) + \frac{\theta_{i+k} - t}{\theta_{i+k} - \theta_{i+1}} N_{i+1}^{k-1}(t) \; .$$

If we are interested in open curves, we will assume that N_i^k is defined on a knot sequence

$$\theta_0 = \cdots = \theta_{k-1}, \theta_k, ..., \theta_n, \theta_{n+1} = \cdots = \theta_{n+k}. \tag{2}$$

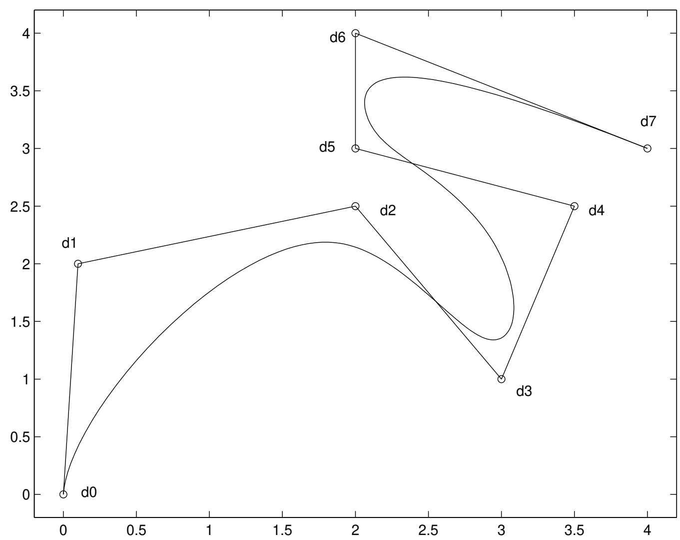

Figure 1: open NURBS curve with $n = 7$, $k = 4$ $\mathbf{T} = (0, 0, 0, 0, 0.2, 0.4, 0.6, 0.8, 1, 1, 1, 1)$ and $\mathbf{W} = (1.4, 0.5, 1.6, 1.8, 0.7, 1.9, 1.5, 0.9)$.

We will assume furthermore that

$$\theta_0 = 0\,, \quad \text{and} \quad \theta_{n+k} = 1\,.$$

In the sequel, we will denote:

$$\begin{aligned} \mathbf{W} &:= (w_0, ..., w_n) && (3)\\ \mathbf{D} &:= (\mathbf{d}_0, ..., \mathbf{d}_n) && (4)\\ \mathbf{T} &:= (\theta_k, ..., \theta_n). && (5) \end{aligned}$$

An illustration can be found in Fig. 1 where we see an open NURBS curve together with its control polygon composed of the control points.

Remark 1 If we work with closed curves (see Fig. 2), the knot sequence relations in (2) and (3) become

$$\left\{ \begin{array}{lcl} \theta_{n+1} &=& \theta_n + \theta_0 \\ \theta_{n+2} &=& \theta_{n+1} + (\theta_1 - \theta_0) \\ \quad\cdots && \\ \theta_{n+k} &=& \theta_{n+k-1} + (\theta_{k-1} - \theta_{k-2}) \end{array} \right. \qquad (6)$$

$$\mathbf{T} := (\theta_1, ..., \theta_{n-1}). \qquad (7)$$

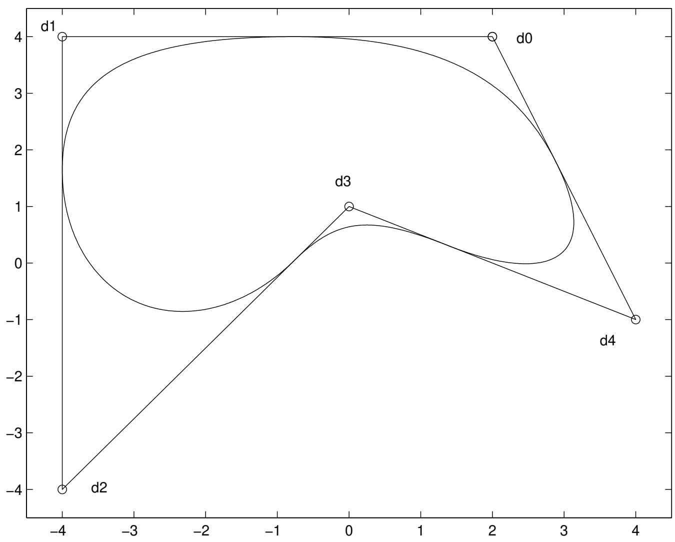

Figure 2: closed NURBS curve with $n = 4$, $k = 3$ $\mathbf{T} = (0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0)$ and $\mathbf{W} = (1.4, 1.2, 0.5, 2.1, 1.1)$.

For the sake of simplicity of the explanation, we will develop our theory for open curves. Note anyway that it can be easily generalized for closed curves as we will illustrate in the numerical results.

2.2 Free knot problem

Suppose we are given a sequence of noisy samples $(t_i, \mathbf{M}_i)$ $i = 0, ..., m$ with $k \ll m$. We want to find the NURBS curve $\mathbf{X}(t) = \mathbf{X}_{\mathbf{W},\mathbf{D},\mathbf{T}}(t)$ which fits these data best in a least square sense. Since we want to find the optimal positions of the knots, we introduce them as variables. That means, we have the following problem:

$$\min_{\mathbf{W},\mathbf{D},\mathbf{T}} \sum_{i=0}^{m} \|\mathbf{X}_{\mathbf{W},\mathbf{D},\mathbf{T}}(t_i) - \mathbf{M}_i\|^2 . \tag{8}$$

This problem is too difficult to solve because of the nonlinear dependence of $\mathbf{X}$ on the parameters $(\mathbf{W}, \mathbf{D}, \mathbf{T})$. Furthermore, we need to add some constraints about the positivity of the weights. In the next section, we will show how to simplify this problem.

2.3 Brief recall of the fixed knot problem

If we are given a knot sequence $\mathbf{T}$, then the problem

$$\min_{\mathbf{W},\mathbf{D}} \sum_{i=0}^{m} \|\mathbf{X}_{\mathbf{W},\mathbf{D},\mathbf{T}}(t_i) - \mathbf{M}_i\|^2$$

will be refered to as the fixed knot problem. It has been investigated for example in [4] where it is shown to be equivalent to solving a linear system:

$$(A + \lambda B)\mathbf{y} = \lambda \mathbf{r} , \tag{9}$$

where A and B are given in block structure:

$$A := \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \; B := \begin{bmatrix} 0 & 0 \\ 0 & \tilde{B} \end{bmatrix}$$

$$A_{rs} := \begin{bmatrix} \Sigma\overline{N}^i_{00}\mathbf{Z}^{rs}_i & \cdots & \Sigma\overline{N}^i_{0n}\mathbf{Z}^{rs}_i \\ \vdots & & \vdots \\ \Sigma\overline{N}^i_{n0}\mathbf{Z}^{rs}_i & \cdots & \Sigma\overline{N}^i_{nn}\mathbf{Z}^{rs}_i \end{bmatrix} \tag{10}$$

$$\mathbf{Z}_i^{(1,1)} := \mathbf{I} - c_i\mathbf{M}_i\mathbf{M}_i^T \tag{11}$$
$$\mathbf{Z}_i^{(1,2)} := c_i\mathbf{M}_i \tag{12}$$
$$\mathbf{Z}_i^{(2,1)} := c_i\mathbf{M}_i^T \tag{13}$$
$$\mathbf{Z}_i^{(2,2)} := 1 - c_i \tag{14}$$

$$\tilde{B} := \begin{bmatrix} \Sigma\overline{N}^i_{00} & \cdots & \Sigma\overline{N}^i_{0n} \\ \vdots & & \vdots \\ \Sigma\overline{N}^i_{n0} & \cdots & \Sigma\overline{N}^i_{nn} \end{bmatrix} \tag{15}$$

$$
\begin{aligned}
\mathbf{y} &:= [\bar{\mathbf{d}}_0, ..., \bar{\mathbf{d}}_n, w_0, ..., w_n]^T \\
\mathbf{r} &:= [\mathbf{0}, ..., \mathbf{0}, \Sigma N_0^k(t_i), ..., \Sigma N_n^k(t_i)]^T \\
\mathbf{I} &:= \text{identity matrix of order } 3 \\
\bar{\mathbf{d}}_i &:= [w_i d_{ix}, w_i d_{iy}, w_i d_{iz}] \\
\overline{N}_{pq}^i &:= N_p^k(t_i) N_q^k(t_i) \\
c_i &:= 1/(1 + \mathbf{M}_i^2).
\end{aligned}
$$

In all these expressions, Σ is understood to be $\sum_{i=0}^m$ and λ is a positive constant which should be chosen large enough (see [4]) in order to ensure positivity of the weights.

3 Solving the free knot problem

The choice of the knot vector $\mathbf{T}$ has a large influence on the quality of the curve fitting result. It is well known that a bad placement of the knots may lead to overshooting effects that distort the shape of the curve. In this section, we describe our method of curve fitting that involves the determination of optimal knot positions.

3.1 Preparing the problem for nonlinear optimization

In the following, we set up the problem such that nonlinear optimization methods can be applied. According to section 2.3, for a given knot $\mathbf{T}$, we can solve the subproblem (9) in order to determine the corresponding weights $\mathbf{W}$ and the control points $\mathbf{D}$. In other words $\mathbf{W}$ and $\mathbf{D}$ are functions of $\mathbf{T}$ i.e.

$$(\mathbf{W}, \mathbf{D}) = (\mathbf{W}(\mathbf{T}), \mathbf{D}(\mathbf{T}))\,.$$

Problem (8) is therefore simplified into:

$$\min_{\mathbf{T}} \sum_{i=0}^{m} \|\mathbf{X}_{\mathbf{W}(\mathbf{T}),\mathbf{D}(\mathbf{T}),\mathbf{T}}(t_i) - \mathbf{M}_i\|^2\,. \tag{16}$$

From now on, we will write only $\mathbf{X}_{\mathbf{T}}(t_i)$ instead of $\mathbf{X}_{\mathbf{W}(\mathbf{T}),\mathbf{D}(\mathbf{T}),\mathbf{T}}(t_i)$ in order to simplify the notation. We have then

$$\min_{\mathbf{T}} \sum_{i=0}^{m} \|\mathbf{X}_{\mathbf{T}}(t_i) - \mathbf{M}_i\|^2\,. \tag{17}$$

This formulation still allows the occurrence of a knot sequence that is not increasing. Therefore, we will modify this problem so that only knots with $\theta_k \leq \theta_{k+1} \leq \cdots \leq \theta_n$ will happen. By denoting:

$$
\begin{aligned}
&\mathbf{X}_{\mathbf{T}}(t) = (X_{\mathbf{T},x}(t), X_{\mathbf{T},y}(t), X_{\mathbf{T},z}(t)) \text{ and} \\
&\mathbf{M}_i = (M_{ix}, M_{iy}, M_{iz})\,,
\end{aligned}
$$

we have

$$\min_{\mathbf{T}} \sum_{i=0}^{m} (X_{\mathbf{T},x}(t_i) - M_{ix})^2 + (X_{\mathbf{T},y}(t_i) - M_{iy})^2 + (X_{\mathbf{T},z}(t_i) - M_{iz})^2 . \tag{18}$$

By defining

$$\begin{cases} S_{3i}(\mathbf{T}) & := & X_{\mathbf{T},x}(t_i) - M_{ix} \\ S_{3i+1}(\mathbf{T}) & := & X_{\mathbf{T},y}(t_i) - M_{iy} \\ S_{3i+2}(\mathbf{T}) & := & X_{\mathbf{T},z}(t_i) - M_{iz} , \end{cases}$$

we obtain

$$\min_{\mathbf{T}} \sum_{i=0}^{3m+2} S_i^2(\mathbf{T}). \tag{19}$$

We introduce now the function

$$R(x) := \begin{cases} 0 & \text{if} \quad x > 0 \\ (-x)^3 & \text{if} \quad x \leq 0 , \end{cases}$$

and we define

$$R(\mathbf{T}) := R(\theta_{k+1} - \theta_k) + R(\theta_{k+2} - \theta_{k+1}) + \cdots + R(\theta_n - \theta_{n-1}) . \tag{20}$$

Instead of (19), we will consider

$$\min_{\mathbf{T}} \sum_{i=0}^{3m+2} [S_i(\mathbf{T}) + \alpha R(\mathbf{T})]^2 , \tag{21}$$

where α is a very large positive number. To understand the relationship between (19) and (21), we note the following two properties of equation (21).

- If we have $\theta_k \leq \theta_{k+1} \leq ... \leq \theta_n$, then $\theta_{k+r} - \theta_{k+r-1} \geq 0$ for all $r = 1, ..., n-k$ and therefore $R(\mathbf{T}) = 0$. Thus,

$$\sum_{i=0}^{3m+2} [S_i(\mathbf{T}) + \alpha R(\mathbf{T})]^2 = \sum_{i=0}^{3m+2} S_i^2(\mathbf{T}) .$$

- If there is some r such that $\theta_{k+r} < \theta_{k+r-1}$, then

  $R(\theta_{k+r} - \theta_{k+r-1}) > 0$ and so $R(\mathbf{T})$ is nonzero. Because of our assumption that α is a very large number, we can expect that $\sum_{i=0}^{3m+2} [S_i(\mathbf{T}) + \alpha R(\mathbf{T})]^2$ is also very large.

Since we are searching for the minimum of $\sum_{i=0}^{3m+2} [S_i(\mathbf{T}) + \alpha R(\mathbf{T})]^2$, the preceeding two points show that a $\mathbf{T}$ with $\theta_{k+r} < \theta_{k+r-1}$ can never realize this minimum. That means that the introduction of the penalizing term in (21) avoids those $\mathbf{T}$ with $\theta_{k+r} < \theta_{k+r-1}$. In situation where it is desirable to have $|\theta_i - \theta_{i+1}| > \varepsilon$ for $i = k, ..., n-1$, we replace (20) by

$$R(\mathbf{T}) := R(\theta_{k+1} - \theta_k - \varepsilon) + R(\theta_{k+2} - \theta_{k+1} - \varepsilon) + ... + R(\theta_n - \theta_{n-1} - \varepsilon) . \tag{22}$$

3.2 Nonlinear optimization

By introducing

$$r_i(\mathbf{T}) := S_i(\mathbf{T}) + \alpha R(\mathbf{T}) \quad \text{and} \quad K := 3m + 2 \ , \tag{23}$$

the curve fitting problem has the form of a usual nonlinear least square problem:

$$\min_{\mathbf{T}} \sum_{i=0}^{K} [r_i(\mathbf{T})]^2 \,. \tag{24}$$

Such a problem can be solved by nonlinear least square solvers like Levenberg-Marquardt and Gauss-Newton (see [12, 6]). Note that for each evaluation of the function $r_i(\mathbf{T})$, we need to solve the subproblem (9) in order to know the corresponding $\mathbf{W}(\mathbf{T})$, $\mathbf{D}(\mathbf{T})$. We note that the order of the linear system (9) is small. It does not depend on the number of data points. It depends exclusively on the degree n of the NURBS curves. Furthermore, taking into account that the support of N_i^k is $[\theta_i, \theta_{i+k})$, we conclude that the matrices in (10) are banded. More precisely, we have:

$$\forall \beta_i \quad \sum_{i=0}^{m} \overline{N}_{pq} \beta_i = 0 \quad \text{for } |p - q| \geq k.$$

As a consequence, the matrices are sparse and therefore we only need to compute a few entries. On the other hand, we must note that the computation of one entry of this system involves all data points. The remedy to that problem is to assemble the following matrix and vector

$$F := \begin{bmatrix} I - c_0 \mathbf{M}_0 \mathbf{M}_0^T \\ I - c_1 \mathbf{M}_1 \mathbf{M}_1^T \\ \dots \\ I - c_m \mathbf{M}_m \mathbf{M}_m^T \end{bmatrix} , \qquad \mathbf{c} := \begin{bmatrix} c_0 \\ c_1 \\ \dots \\ c_m \end{bmatrix} \tag{25}$$

only once and store them in arrays so that they do not need to be recomputed in subsequent computations. Note that F and $\mathbf{c}$ are independent of $\mathbf{T}$. They depend only on the initial data points. The only expressions which need to be updated in each iteration of the nonlinear optimization are the values of $N_l^j(t_i)$ which are mostly zero except for some few values. They can be computed recursively in a fast way (see [1]).

3.3 Numerical results

The numerical results in this paper have been computed with the Levenberg-Marquardt algorithm. Note that this algorithm is iterative and so it needs some initial guess. The initial guess that we have taken here is equidistant knots in which the approximating curve is very far from the true curve. The first numerical test that we perform is the reconstruction of W-shaped curve. We have 75 data which were added with random noise of amplitude 0.1. The curve is reconstructed with the help of the formerly described algorithm. The overall time for the reconstruction is 8 seconds. In Figure 3 we see a graphical illustration of the data, the initial curve and the reconstructed curve. The second test is a free-form curve. The time needed for the reconstruction is like in the first test. A graphical illustration can be found in Figure 4.

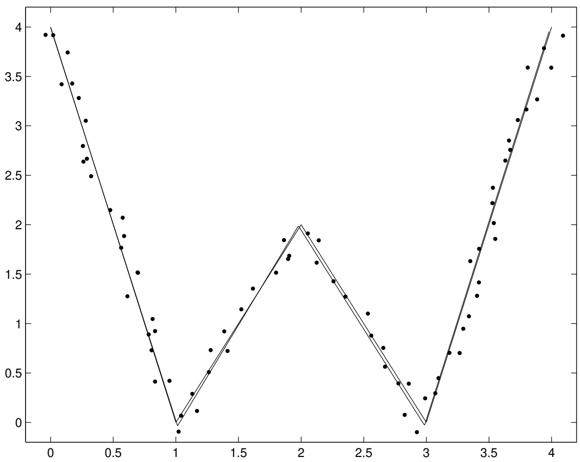

Figure 3: Initial curve, reconstructed curve and 75 samples with noise amplitude=0.1

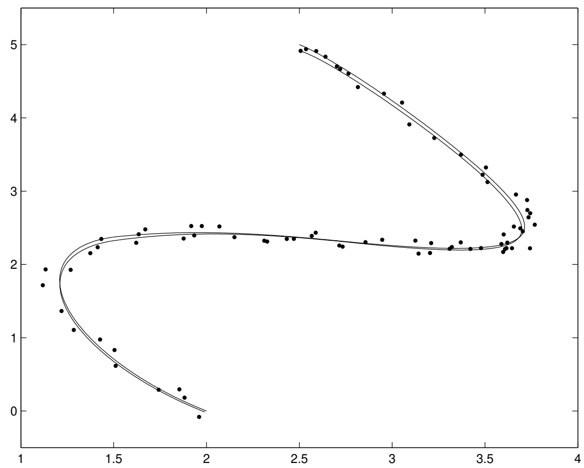

Figure 4: Initial curve, reconstructed curve and 75 samples with noise amplitude=0.1

4 Parallel realization

When the number of data becomes very large, the performance of the described algorithm will be slow. For this reason we considered the parallel implementation of the algorithm

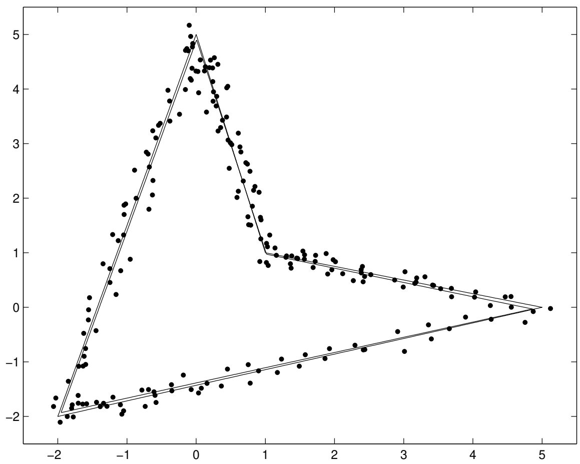

Figure 5: Initial closed curve, reconstructed closed curve and 75 noisy samples

which will be described in this section. All the numerical results that will follow were obtained from the CLiC (Chemnitzer Linux Cluster) supercomputer ([3]) and the parallel program was implemented by using MPI (Message Passing Interface, see [13, 14, 8]).

4.1 Way of parallelizing

Let us first note that in each iteration of the Levenberg-Marquardt algorithm (see [19]) we have to compute the Jacobi matrix $\mathbf{J}(\mathbf{T})$. Furthermore, we need to compute the matrix F and the vector $\mathbf{c}$ in relation (25). The computation of these three arrays is the most expensive operations of our method. Therefore we conclude that parallelizing the problem of curve reconstruction basically means assembling these matrices in parallel. A more detailed description of parallelizing the Levenberg-Marquardt algorithm is planned in [16]. For briefness, we only describe the way we do it with the matrix $\mathbf{J}(\mathbf{T})$. The arrays F and $\mathbf{c}$ can be assembled in a similar way. The components of the unknown $\mathbf{T}$ will be denoted by τ_i i.e. $\mathbf{T} = (\tau_1, ..., \tau_l)$ where $l := n - k + 1$. The entries of $\mathbf{J}(\mathbf{T})$ are given by:

$$J_{ij}(\mathbf{T}) = \frac{\partial r_i(\mathbf{T})}{\partial \tau_s}\,.$$

The Jacobian matrix $\mathbf{J}(\mathbf{T})$ is of size $(3m + 3) \times l$. That means that it has in practice very large number of rows and a few columns. When the number of points become very large, it is very costly to assemble and store this matrix. We can distribute the entries of $\mathbf{J}(\mathbf{T})$ to the processors. In other words, we partition the matrix $\mathbf{J}(\mathbf{T})$ and each processor computes the corresponding part of $\mathbf{J}(\mathbf{T})$. If we suppose that the number of processors is κ, a possible partitioning of the matrix $\mathbf{J}(\mathbf{T})$ can be found in Fig. 6.

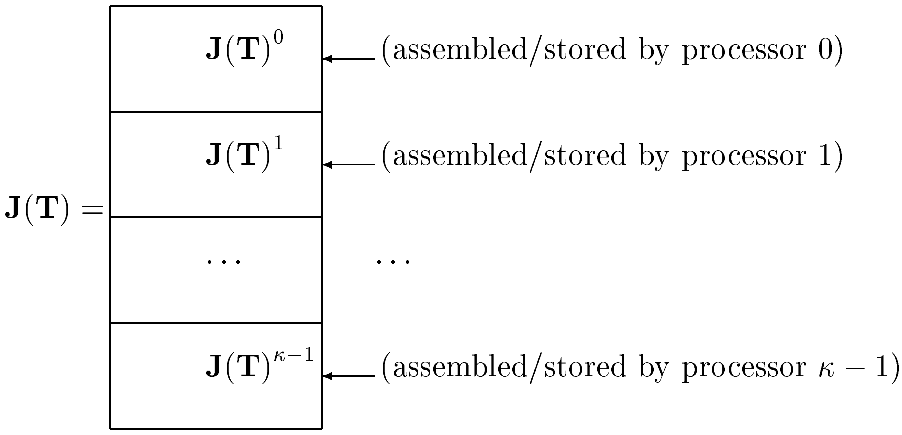

Figure 6: Distribution of the matrix $\mathbf{J}(\mathbf{T})$ to the processors

4.2 Load balancing

In this section we describe our method to achieve load balancing. We want in fact that all the processors execute approximately the same amount of computational work. If the number of processors κ is a multiple of the number of rows $3m + 3$, then the process is easy because each processor treats $L = \frac{3m+3}{\kappa}$ rows. Otherwise, we introduce

$$m_0 := \left\lfloor \frac{3m+3}{\kappa} \right\rfloor, \qquad m_1 := 3m + 3 - m_o\kappa$$

(where $\lfloor x \rfloor$ stands for the largest integer which is smaller or equal to x).
For processor $p = 0, ..., \kappa - 1$, the number of rows that it assembles and stores is:

$$L(p) := \begin{cases} m_0 & \text{if} \quad p \geq m_1 \\ m_0 + 1 & \text{if} \quad p < m_1 \end{cases}$$

We notice that $L(p)$ is either m_0 or $m_0 + 1$ which are approximately the same in practical situations (i.e. m_0 large). That means that load balancing can still be achieved in cases where $3m + 3$ is not a multiple of the number of processors.
As an illustration: if we have $3m + 3 = 123,025$ data points and we have $\kappa = 9$ processors, then $L(p) = 13669$ if $p \geq 4$ and $L(p) = 13670$ otherwise.

4.3 Parallel performance

In this section, we evaluate the performance of our parallel program. First, we investigate the speedup of our algorithm. We recall the definition (see [15]) of speedup for number of processors κ to be:

$$S(\kappa) := \frac{\text{runtime of serial program}}{\text{runtime of parallel program with}\,\kappa\,\text{processors}}$$

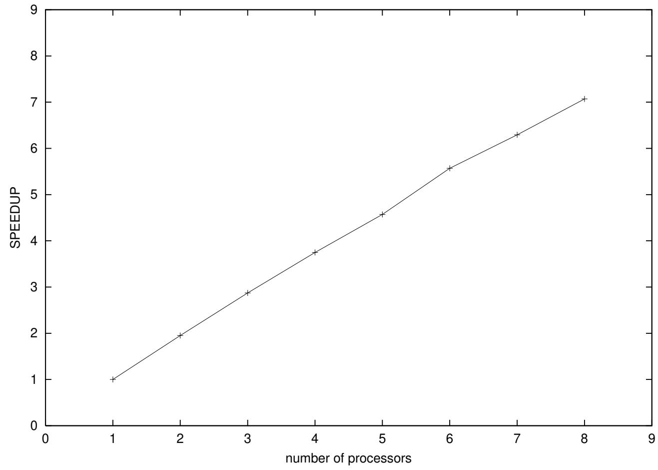

Figure 7: Speedup

In Fig. 7, we plot the speedup of our parallel algorithm. We can notice that we have a speedup which is practically linear. In other words, if we multiply the number of processors by two, then the time for running the algorithm is also twice faster.
Secondly, we want to investigate the efficiency (see [15]) which is defined as

$$E(\kappa) := \frac{S(\kappa)}{\kappa}$$

We know (see [14]) that $0 < E(\kappa) \leq 1$: if the efficiency $E(\kappa)$ approaches 0 then the parallel program is said to be bad because it reflects a slowdown. Contrarily, an efficiency of 1 demonstrate a good performance. In Table 1, we gather the numerical results for the efficiency. As we can see, our algorithm can be categorized as having good performance because the efficiency is always close to 1 .

nb. processors	runtime parallel	efficiency
1	304.31 sec	1.00000
2	155.92 sec	0.975871
3	105.91 sec	0.957776
4	81.20 sec	0.936849
5	66.58 sec	0.914051
6	54.64 sec	0.928139

Table 1: Efficiency

In Table 2, we collect some results of numerical experiments for different numbers of data with three numbers of processors. It shows more precisely the runtime of our algorithm in practice.

number of data	64 processors	32 processors	16 processors
500,000	17.18 sec	28.07 sec	50.21 sec
600,000	19.81 sec	29.70 sec	54.76 sec
700,000	22.50 sec	35.47 sec	63.78 sec
800,000	26.26 sec	40.70 sec	76.85 sec
900,000	29.62 sec	44.85 sec	81.71 sec
1,000,000	31.82 sec	50.70 sec	93.72 sec

Table 2: runtime in second

4.4 Iteration vs. error

The next test consists in investigating the error after each iteration. This test is for the W-shaped curve in which the exact inner knot position is $\mathbf{T} = (0.333333, 0.5, 0.666666)$. In Table 3, we see the value of $\mathbf{T}$ for each iteration and the corresponding error. We found that the number of iterations needed in practice to achieve a satisfactory accuracy. Note also that the reconstructed curve is already graphically satisfactory after iteration 4.

Iter	$\mathbf{T}(0)$	$\mathbf{T}(1)$	$\mathbf{T}(2)$	Error
1	0.25000	0.50000	0.75000	0.055556
2	0.26471	0.50124	0.71852	0.040575
3	0.29438	0.50293	0.69673	0.023980
4	0.31266	0.50099	0.67307	0.009359
5	0.32722	0.49908	0.66601	0.002561
6	0.33331	0.49992	0.66668	0.000037

Table 3: Error for each iteration.

4.5 Initial guess

The number of the iterations needed for this method depend on the initial guess that we take. That is again due to the fact that this is an iterative method. Although two different initial guesses give the same results, they may need different time to run the algorithm because more iterations mean a longer time of execution. Two possible initial guesses are: equidistant knot sequence and chord length knot sequence. The latter is meant in the sense that the interval $[0, 1]$ is divided into subintervals in such a way that each corresponding curve part is proportional to the length of the subinterval. Sometimes the first initial guess is good. For instance, the W-shaped curve (see Fig. 3) needs 6 iterations for equidistant initial guess. But it needs as many as 11 iterations for chord length initial guess to have the same results. The case of the curve in Fig. 4 is completely opposite to that. It needs 4 iterations for equidistant initial guess and 3 iterations for chord length initial guess. But as far as the final results are considered, both initial guesses give the same result. The difference can be found only in the number of iterations.

5 Future work

In this work, we have only focused on curve reconstruction. The case of surfaces is more interesting because it has exciting applications in reverse engineering. Our future work will consist in extending our algorithm for surfaces.

References

[1] C. de Boor, A practical guide to splines, Springer (New York, 1978).

[2] J. Corney, 3D modeling with ACIS kernel and toolkit, John Wiley & sons (Chichester, 1997).

[3] Chemnitzer Linux Cluster. http://www.tu-chemnitz.de/urz/

[4] B. Elsässer, Approximation mit rationalen B-Spline Kurven und Flächen, Ph.D. thesis, Department of mathematics, Darmstadt, 1998.

[5] G. Farin, Curves and surfaces for computer aided geometric design, Academic Press, 2nd edition (Boston, 1990).

[6] R. Fletcher, Practical methods of optimization, John Wiley & Sons, 2nd edition (Chichester, 1987).

[7] J. Hoschek and D. Lasser, Grundlagen der geometrischen Datenverarbeitung, Teubner (Stuttgart, 1989).

[8] P. K. Jimack and N. Touheed, An introduction to MPI for computational mechanics, in: Parallel and Distributed Processing for Computational Mechanics: Systems and Tools, ed. B.H.V. Topping (Saxe-Coburg, 1999) 24-25.

[9] D. Jupp, Approximation to data by splines with free knots, SIAM J. Numer. Anal. 15, No. 2(1978) 328-343.

[10] P. Laurent-Gengoux and M. Mekhilef, Optimization of a NURBS representation, Computer Aided Design 25, No. 11 (1993) 699-710.

[11] J. Neider, T. Davis and M. Woo, OpenGL programming guide, Addison-Wesley publishing company (Reading, 1994).

[12] J. Nocedal and S. Wright, Numerical Optimization, Springer Series in Operation Research (New York, 1999).

[13] P. S. Pacheco, A user's guide to MPI, Technical report, Department of mathematics, University of San Fransisco, California, 1998.

[14] P. S. Pacheco, Parallel programming with MPI, Morgan Kaufmann (San Francisco, 1997).

[15] S. Ragsdale, Parallel programming, McGraw-Hill (New York, 1991).

[16] M. Randrianarivony and G. Brunnett, Parallel implementation of surface reconstruction from noisy samples, SFB preprint 02-16, Technische Universität Chemnitz, 2002.

[17] T. Schütze, Diskrete Quadratmittelapproximation durch Splines mit freien Knoten, Ph.D. thesis, Mathematics faculty, Dresden, 1998.

[18] H. Schwetlick and T. Schütze, Least squares approximation by splines with free knots, BIT 35, No. 3 (1995) 361-384.

[19] J. Z. Zhang and L.H. Chen, Nonmonotone Levenberg-Marquardt algorithms and their convergence analysis, Journal of optimization theory and applications 92, No. 2 (1997) 393-418.

Other titles in the SFB393 series:

01-01 G. Kunert. Robust local problem error estimation for a singularly perturbed problem on anisotropic finite element meshes. January 2001.

01-02 G. Kunert. A note on the energy norm for a singularly perturbed model problem. January 2001.

01-03 U.-J. Görke, A. Bucher, R. Kreißig. Ein Beitrag zur Materialparameteridentifikation bei finiten elastisch-plastischen Verzerrungen durch Analyse inhomogener Verschiebungsfelder mit Hilfe der FEM. Februar 2001.

01-04 R. A. Römer. Percolation, Renormalization and the Quantum-Hall Transition. February 2001.

01-05 A. Eilmes, R. A. Römer, C. Schuster, M. Schreiber. Two and more interacting particles at a metal-insulator transition. February 2001.

01-06 D. Michael. Kontinuumstheoretische Grundlagen und algorithmische Behandlung von ausgewählten Problemen der assoziierten Fließtheorie. März 2001.

01-07 S. Beuchler. A preconditioner for solving the inner problem of the p-version of the FEM, Part II - algebraic multi-grid proof. March 2001.

01-08 S. Beuchler, A. Meyer. SPC-PM3AdH v 1.0 - Programmer's Manual. March 2001.

01-09 D. Michael, M. Springmann. Zur numerischen Simulation des Versagens duktiler metallischer Werkstoffe (Algorithmische Behandlung und Vergleichsrechnungen). März 2001.

01-10 B. Heinrich, S. Nicaise. Nitsche mortar finite element method for transmission problems with singularities. March 2001.

01-11 T. Apel, S. Grosman, P. K. Jimack, A. Meyer. A New Methodology for Anisotropic Mesh Refinement Based Upon Error Gradients. March 2001.

01-12 F. Seifert, W. Rehm. (Eds.) Selected Aspects of Cluster Computing. March 2001.

01-13 A. Meyer, T. Steidten. Improvements and Experiments on the Bramble–Pasciak Type CG for mixed Problems in Elasticity. April 2001.

01-14 K. Ragab, W. Rehm. CHEMPI: Efficient MPI for VIA/SCI. April 2001.

01-15 D. Balkanski, F. Seifert, W. Rehm. Proposing a System Software for an SCI-based VIA Hardware. April 2001.

01-16 S. Beuchler. The MTS-BPX-preconditioner for the p-version of the FEM. May 2001.

01-17 S. Beuchler. Preconditioning for the p-version of the FEM by bilinear elements. May 2001.

01-18 A. Meyer. Programmer's Manual for Adaptive Finite Element Code SPC-PM 2Ad. May 2001.

01-19 P. Cain, M.L. Ndawana, R.A. Römer, M. Schreiber. The critical exponent of the localization length at the Anderson transition in 3D disordered systems is larger than 1. June 2001

01-20 G. Kunert, S. Nicaise. Zienkiewicz-Zhu error estimators on anisotropic tetrahedral and triangular finite element meshes. July 2001.

01-21 G. Kunert. A posteriori H^1 error estimation for a singularly perturbed reaction diffusion problem on anisotropic meshes. August 2001.

01-22 A. Eilmes, Rudolf A. Römer, M. Schreiber. Localization properties of two interacting particles in a quasi-periodic potential with a metal-insulator transition. September 2001.

01-23 M. Randrianarivony. Strengthened Cauchy inequality in anisotropic meshes and application to an a-posteriori error estimator for the Stokes problem. September 2001.

01-24 Th. Apel, H. M. Randrianarivony. Stability of discretizations of the Stokes problem on anisotropic meshes. September 2001.

01-25 Th. Apel, V. Mehrmann, D. Watkins. Structured eigenvalue methods for the computation of corner singularities in 3D anisotropic elastic structures. October 2001.

01-26 P. Cain, F. Milde, R. A. Römer, M. Schreiber. Use of cluster computing for the Anderson model of localization. October 2001. Conf. on Comp. Physics, Aachen (2001).

01-27 P. Cain, F. Milde, R. A. Römer, M. Schreiber. Applications of cluster computing for the Anderson model of localization. October 2001. Transworld Research Network for a review compilation entitled "Recent Research Developments in Physics", (2001).

01-28 X. W. Guan, A. Foerster, U. Grimm, R. A. Römer, M. Schreiber. A supersymmetric Uq[(osp)(2—2)]-extended Hubbard model with boundary fields. October 2001.

01-29 K. Eppler, H. Harbrecht. Numerical studies of shape optimization problems in elasticity using wavelet-based BEM. November 2001.

01-30 A. Meyer. The adaptive finite element method - Can we solve arbitrarily accurate? November 2001.

01-31 H. Harbrecht, S. Pereverzev, R. Schneider. An adaptive regularization by projection for noisy pseudodifferential equations of negative order. November 2001.

01-32 G. N. Gatica, H. Harbrecht, R. Schneider. Least squares methods for the coupling of FEM and BEM. November 2001.

01-33 Th. Apel, A.-M. Sändig, S. I. Solov'ev. Computation of 3D vertex singularities for linear elasticity: Error estimates for a finite element method on graded meshes. December 2001.

02-01 M. Pester. Bibliotheken zur Entwicklung paralleler Algorithmen - Basisroutinen für Kommunikation und Grafik. Januar 2002.

02-02 M. Pester. Visualization Tools for 2D and 3D Finite Element Programs - User's Manual. January 2002.

02-03 H. Harbrecht, M. Konik, R. Schneider. Fully Discrete Wavelet Galerkin Schemes. January 2002.

02-04 G. Kunert. A posteriori error estimation for convection dominated problems on anisotropic meshes. March 2002.

02-05 H. Harbrecht, R. Schneider. Wavelet Galerkin Schemes for 3D-BEM. February 2002.

02-06 W. Dahmen, H. Harbrecht, R. Schneider. Compression Techniques for Boundary Integral Equations - Optimal Complexity Estimates. April 2002.

02-07 S. Grosman. Robust local problem error estimation for a singularly perturbed reaction-diffusion problem on anisotropic finite element meshes. May 2002.

02-08 M. Springmann, M. Kuna. Identifikation schädigungsmechanischer Materialparameter mit Hilfe nichtlinearer Optimierungsverfahren am Beispiel des Rousselier Modells. Mai 2002.

02-09 S. Beuchler, R. Schneider, C. Schwab. Multiresolution weighted norm equivalences and applications. July 2002.

02-10 Ph. Cain, R. A. Römer, M. E. Raikh. Renormalization group approach to energy level statistics at the integer quantum Hall transition. July 2002.

02-11 A. Eilmes, R. A. Römer, M. Schreiber. Localization properties of two interacting particles in a quasiperiodic potential with a metal-insulator transition. July 2002.

02-12 M. L. Ndawana, R. A. Römer, M. Schreiber. Scaling of the Level Compressibility at the Anderson Metal-Insulator Transition. September 2002.

02-13 Ph. Cain, R. A. Römer, M. E. Raikh. Real-space renormalization group approach to the quantum Hall transition. September 2002.

02-14 A. Jellal, E. H. Saidi, H. B. Geyer, R. A. Römer. A Matrix Model for $\nu_{k_1 k_2} = \frac{k_1+k_2}{k_1 k_2}$ Fractional Quantum Hall States. September 2002.

The complete list of current and former preprints is available via
`http://www.tu-chemnitz.de/sfb393/preprints.html`.