Greedy algorithms for learning via exponential-polynomial splines

R. Campagna *1, S. De Marchi †2, E. Perracchione ‡3, and G. Santin §4

1 University of Campania “L. Vanvitelli”, Italy
2 University of Padova, Italy
3 CNR-SPIN, University of Genova, Italy
4 DIGIS, Bruno Kessler Foundation, Italy

February 5, 2022

Abstract

Kernel-based schemes are state-of-the-art techniques for learning by data. In this work we extend some ideas about kernel-based greedy algorithms to exponential-polynomial splines, whose main drawback consists in possible overfitting and consequent oscillations of the approximant. To partially overcome this issue, we introduce two algorithms which perform an adaptive selection of the spline interpolation points based on the minimization either of the sample residuals (f-greedy), or of an upper bound for the approximation error based on the spline Lebesgue function (λ-greedy). Both methods allow us to obtain an adaptive selection of the sampling points, i.e. the spline nodes. However, while the f-greedy selection is tailored to one specific target function, the λ-greedy algorithm is independent of the function values and enables us to define a priori optimal interpolation nodes.

1 Introduction

Scattered data interpolation is one of the most investigated topics in the field of numerical analysis, and it is successfully used for many applications. As a consequence, many methods have been developed, including interpolation with polynomials of total degree (see e.g. [6]), splines approximation [10] [15] [16],
with its recent developments in the context of machine learning [3, 29], and kernel-based methods (refer e.g. to [19, 30]).

More recently, the so-called Exponential-Polynomial Splines (EPS) have been introduced with the main purpose of approximating univariate multieponential decay functions with a smoothing effect [9, 11, 12]. Such a smoothing strategy is implemented by considering a regularization parameter.

In this paper we are interested in the design of appropriate sampling strategies for EPS interpolation. Namely, assuming to be given either only an input space discretization, or a dataset of input points and corresponding function evaluations, we aim at selecting a small subset of approximation points to be used to construct the EPS interpolant. We consider in particular incremental methods that, given an initial set of samples, construct an EPS interpolant by iteratively selecting a new point at each iteration. The iterative rule is dictated by greedy methods (see [28]), which have been investigated e.g. for kernel methods (see e.g. [20, 22, 25, 31]) and lead to sparse models which turn out to be helpful in many applications, see e.g. [18]. This iterative selection is a convenient proxy for the optimal selection of the sampling points, which is in turn usually an extremely computationally demanding procedure.

Since the greedy selection is based on the maximization of an error indicator, to use such schemes in the context of EPS we first study pointwise error bounds. We are able to bound the pointwise error thanks to the definition of the cardinal form of the EPS interpolant that then allows us to introduce the Lebesgue function and constant [7, 8]. The latter are known to be stability indicators for polynomial bases; see e.g. [1, 2, 4, 5, 6, 17]. Based on this new error indicator, we define an algorithm for selecting data-independent points for EPS.

Furthermore, we propose a second extraction strategy that takes into account also the function values. This kind of approach is usually more expensive, but it allows us to select points that are tailored to one specific target function, and thus are usually able to better resolve local features such as steep gradients or oscillations.

In both cases, we numerically explore the behavior of the node distribution for the spline basis, and we test our findings under different perspectives.

The paper is organized as follows. In Section 2 we briefly review the basics of greedy methods and EPS interpolation. Error bounds and adaptive strategies for the greedy selection of the nodes are presented in Section 3. Some numerical experiments are presented and discussed in Section 4 while conclusions with an outline of future works are provided in Section 5.

2 Exponential splines and greedy schemes

In this section we present the main features of EPS and greedy methods.

We consider a function \( f : [a, b] \rightarrow \mathbb{R} \) with \([a, b] \subset \mathbb{R}\) and an associated set of function values \( F := \{ y_i := f(x_i) \}_{i=1}^n \) sampled at a data set \( X := \{ x_i \}_{i=1}^n \), with \( a = x_1 < x_2 < \cdots < x_n = b \). Our goal is to construct an approximation of the unknown function \( f \), and we concentrate on interpolatory schemes, i.e., our
model $I_X$ of $f$ satisfies $I_X(x_i) = y_i$, $1 \leq i \leq n$.

In general terms, given a normed linear space of functions defined in $[a, b] \subset \Omega$, and an associated basis $\{b_j\}_{j=1}^n \subset C([a, b])$, an interpolant $I_X : [a, b] \rightarrow \mathbb{R}$ may be defined as

$$I_X(x) = \sum_{j=1}^n c_j b_j(x), \quad x \in [a, b].$$

Provided that $\{b_j\}_{j=1}^n$ form a Haar system (see e.g. [39]), the matrix $A \in \mathbb{R}^{n \times n}$ with $A_{ij} := b_j(x_i)$, $1 \leq i, j \leq n$, is invertible for any set of interpolation points, and the coefficients $c := [c_1, \ldots, c_n]^T \in \mathbb{R}^n$ of the interpolant may be determined by solving the system

$$Ac = y,$$

where $y := [y_1, \ldots, y_n]^T \in \mathbb{R}^n$. Popular basis functions that meet these requirements are, for instance, Radial Basis Functions (RBFs), or monomials of total degree $n - 1$.

### 2.1 Exponential splines

In this work we focus our attention on a particular basis $\{b_j\}_{j=1}^n$ of splines that was introduced in [11]. In this section we recall the definition of the smoothing spline model introduced in the cited paper and give the basis definition in details.

**Definition 1.** Let $\{(x_i, y_i)\}_{i=1}^n \subset \Omega \times \mathbb{R}$ be given, with $a = x_1$ and $b = x_n$. Let $w_1, \ldots, w_n$ be non zero weights, $\beta > 0$ be a regularization parameter.

For $\alpha > 0$ and $L_2^u := u'' + 2\alpha u' + \alpha^2 u$, we denote as $L_2^*$ the adjoint, and as $E_{4,\alpha} := \{e^{\alpha x}, xe^{\alpha x}, e^{-\alpha x}, xe^{-\alpha x}\}$ the null space of $L_2^* L_2$.

Then the smoothing exponential spline $I_{X,\alpha}(f) : [a, b] \rightarrow \mathbb{R}$ is the solution of the penalized least square problem

$$\min_{c_1, \ldots, c_n} \sum_{i=1}^n w_i \left( y_i - \sum_{j=1}^n c_j \varphi_j(x_i) \right)^2 + \beta \int_a^b \left( \sum_{j=1}^n c_j L_2 \varphi_j(x) \right)^2 dx, \quad (1)$$

with $\varphi_j|_{[x_i, x_{i+1}]} \in E_{4,\alpha}$, where $|A|$ denotes the restriction on a set $A \subset \mathbb{R}$.

In the same paper [11], an optimal basis $\{\varphi_j\}_{j=1}^n$ of exponential B-splines, also referred to as Generalized B-splines (GB-splines), has been defined. The basis functions have the following properties (see fig. 1): are bell-shaped with compact support, identified by 5 nodes, with the blending segments belonging to $E_{4,\alpha}$ and $C^2$-smoothness. The generic basis function $\varphi$ can then be expressed as

$$\varphi(x)|_{[x_i, x_{i+1}]} = \sum_{k=1}^4 b_{i,k} g_k(x), \quad g_k \in E_{4,\alpha}, \quad (2)$$

where $x_i$ with $i = 1, \ldots, n-1$, denotes the left point of the partition element, and $k = 1, \ldots, 4$ denotes the index of the local basis element. Indeed, each function
φ in the form (2) has $4(n - 1)$ degrees of freedom, given by the coefficients $\{b_{i,k}\}$ with $1 \leq i \leq n - 1$, $1 \leq k \leq 4$. Then, to define such a basis of dimension $n$, the nodes vector has to be augmented with two extra nodes before $x_1$ and two others after $x_n$, i.e. an augmented node set as $x_{-1} < x_0 < x_1 = a < \ldots < x_n = b < x_{n+1} < x_{n+2}$ has to be considered. Such extra nodes affect the construction of the so called boundary basis functions: $\varphi_1, \varphi_2$ and $\varphi_{n-1}, \varphi_n$.

Figure 1: An example of GB-spline with segments in the spaces $\mathbb{E}_{4,\alpha}$.

Remark 1. We assume that these four points are fixed, indeed, numerically we observed that, for our scope, the approximation is not very sensitive with respect to their selection.

In this paper we focus on unweighted interpolation, i.e., no smoothing parameter nor weights are considered: we set $\beta = 0$ and $w_i = 1$, $1 \leq i \leq n$. Moreover, the reason why the authors in [11] assume $\alpha > 0$ is due to the fact that they are interested in modeling functions that decay exponentially. In order to force the model decreasing outside, the authors assume also that the segments of the GB-splines falling outside $[x_1, x_n]$ are defined in a two dimensional space $\mathbb{E}_{2,\alpha} := \{e^{-\alpha x}, x e^{-\alpha x}\}$, null space of $L_2$ (for further details see [11]). The global space of the exponential-polynomial splines $\mathcal{E}_{X,\alpha}(\mathbb{E}_{2,\alpha}, \mathbb{E}_{4,\alpha})$ is then defined by gluing local patches defined over each interval such that $\mathcal{E}_{X,\alpha}(\mathbb{E}_{2,\alpha}, \mathbb{E}_{4,\alpha}) \subset C^2([a,b])$.

Here instead, we focus on a more general model, where only $\mathbb{E}_4$ is considered. Then, we assume $\mathcal{E}_{X,\alpha} := \mathcal{E}_{4,\alpha}(\mathbb{E}_4)$ and any of its elements can be expressed using coefficients $b_{i,k}$ as in (2).

Following [11], we use a Bernstein-like basis to represent each segment of the GB-splines $\{\varphi_j\}_{j=1}^n$ (see [11] Appendix for an explicit construction), as follows:

$$\varphi_j(x)|_{[x_i, x_{i+1}]} = \sum_{k=1}^{4} \gamma_{i,j,k} B_k(x - x_j),$$

where $B_k$, $k = 1, \ldots, 4$, are Bernstein like functions, $1 \leq i \leq n - 1$ and $1 \leq j \leq n$. The existence and uniqueness of such a functional space is provided in [11] Theorem 2.1]. The advantage of the GB-spline basis is that the computations can be performed locally in the support of each $\varphi_j$, indeed $\text{supp}(\varphi_j) \subset [x_{j-2}, x_{j+2}]$, $1 \leq j \leq n$. In particular, the global interpolation matrix $\Phi$ with entries given by $\Phi_{ij} := \varphi_j(x_i)$ is nonzero only on the main diagonal and on the two upper and lower diagonals.
Fixed the space of the described exponential splines, any function \( f : [a, b] \to \mathbb{R} \) can now be approximated on \( X := \{x_i\}_{i=1}^n \) via exponential-polynomial splines interpolation of the associated samples \( F = \{y_i = f(x_i)\}_{i=1}^n \), i.e.,
\[
I_{X,\alpha}(f)(x) = \sum_{j=1}^n c_j \varphi_j(x),
\]
with a vector of coefficients \( c := [c_1, \ldots, c_n]^\top \in \mathbb{R}^n \) such that
\[
\Phi c = y,
\]
where \( y := [y_1, \ldots, y_n]^\top \in \mathbb{R}^n \). Observe that we have \( I_{X,\alpha}(f) = f \) for all \( f \in \mathcal{E}_{X,\alpha} \), i.e., every function in \( \mathcal{E}_{X,\alpha} \) is uniquely determined by its values on \( X \). We summarize in Algorithm 1 the steps for computing the EPS interpolant.

Algorithm 1 Pseudo-code for EPS
\begin{enumerate}
\item Definition of the function space \( \mathcal{E}_4 \) by setting \( \alpha \).
\item Definition of the augmented nodes to define the boundary basis functions.
\item Definition of the GB-spline basis functions in Bernstein-like basis, \( \{\varphi_j\}_{j=1}^n \).
\item Computation of the collocation matrix as in (5) and solution of the collocation system.
\end{enumerate}

Before investigating the use of greedy schemes for EPS, we briefly recall the main ideas behind greedy techniques.

2.2 Greedy schemes
Given \( X \) and \( F \), the main goal of the greedy algorithms consists in selecting a suitable subset \( \tilde{X} \subset X \) so that the greedy interpolant is constructed on a reduced number of data producing an approximation of \( I_X \). Such iterative algorithms belong essentially to two classes:

- Residual-based greedy schemes: the set \( \tilde{X} \) is constructed taking into account the function values \( F \).
- Error-based greedy methods: the set \( \tilde{X} \) is built independently of the function values \( F \).

The general iterative rules for these two algorithms are summarized in Table 1, where \( \lambda \) denotes a pointwise approximation error independent of the function values. Both methods will be investigated in the next section for the special case of EPS.

3 Greedy schemes for EPS
In this section we first recall a simple residual-based greedy scheme, that is known as \( f \)-greedy in literature (see [20]), and that can be easily used with any
Greedy Method | Iterative Rule
---|---
Residual-based | $x^* = \arg\max_{x \in \Omega \setminus X} |f(x) - I_{X,\alpha}(f)(x)|$
Error-based | $x^* = \arg\max_{x \in \Omega \setminus X} \lambda(x)$

Table 1: Iterative rules for residual and error-based greedy strategies.

approximation basis. On the other hand, error-based greedy schemes need to be tailored for the considered basis, and we will discuss their derivation in the case of EPS.

### 3.1 Residual-based greedy selection

As already mentioned, $f$-greedy schemes are quite straightforward to extend to any kind of basis. Precisely, we consider an initial (training) set of sorted data $\tilde{X} = \{x_i\}_{i=1}^q \subset X$, with $x_1 = a, x_q = b$ and we also keep the augmented nodes fixed. Then, given $F$ and a fixed tolerance $\tau$, the residual-based greedy scheme for exponential splines is summarized in Algorithm 2.

**Algorithm 2 Pseudo-code for the $f$-greedy algorithm**

1: Take an initial set of sorted data $\tilde{X} = \{x_i\}_{i=1}^q \subset X; \; x_1 = a, x_q = b$ and $q \geq 2$.
2: Compute an initial interpolant $I_{\tilde{X},\alpha}(f)$ as in (4).
3: While $\max_{x_i \in X \setminus \tilde{X}} |f(x_i) - I_{\tilde{X},\alpha}(f)(x_i)| > \tau$:
   1. Define $x^* = \arg\max_{x_i \in X \setminus \tilde{X}} |f(x_i) - I_{\tilde{X},\alpha}(f)(x_i)|$.
   2. Set $\tilde{X} = \tilde{X} \cup \{x^*\}$ and sort $\tilde{X}$.
   3. Compute $I_{\tilde{X},\alpha}(f)$ as in (4).

The result of the $f$-greedy scheme is thus a set of data locations $\hat{X} = \{x_i\}_{i=1}^n$ with $a = x_1 < \ldots < x_n = b$, and the corresponding interpolant $I_{\hat{X},\alpha}(f)$. Since we usually have $n \ll n$, the greedy interpolant $I_{\hat{X},\alpha}(f)$ can be understood as a sparse approximation of $I_{X,\alpha}(f)$.

This scheme is very easy to implement, and additionally the interpolation points are selected adaptively in order to be suited for the particular target function $f$, and they are thus expected to provide an accurate approximation.

On the other hand this adaptivity may backfire. Indeed, in many applications one aims instead at the selection of a set of interpolation points that can be used to approximate $N_f \in \mathbb{N}$ different functions. In this case, the $f$-greedy algorithm should be executed $N_f$ times, and this constitutes a computational drawback. To deal with this scenario, we drive our attention towards error-based greedy schemes.
3.2 Error-based greedy selection

To investigate the error-based greedy selection, we need to introduce a pointwise error bound for EPS interpolation.

3.2.1 Lagrange functions and Lebesgue constant

Given \( \{\varphi_j\}_{j=1}^n \) as in (2), since the associated matrix \( \Phi \) is invertible we may write \( d_{j\ell} := (\Phi^{-1})_{j\ell} \). In this way we have that the functions

\[
\psi_{\ell}(x) := \sum_{j=1}^n d_{j\ell} \varphi_j(x), \quad 1 \leq \ell \leq n,
\]

satisfy the cardinal conditions

\[
\psi_{\ell}(x_i) = \delta_{i\ell} \quad 1 \leq i, \ell \leq n,
\]

i.e., they are a global Lagrange (or cardinal) basis. To see this, just observe that for \( 1 \leq i, \ell \leq n \), it holds true that

\[
\psi_{\ell}(x_i) = \sum_{j=1}^n \varphi_j(x_i) d_{j\ell} = \sum_{j=1}^n \Phi_{ij} (\Phi^{-1})_{j\ell} = (\Phi \cdot \Phi^{-1})_{i\ell} = \delta_{i\ell}, \quad 1 \leq i, \ell \leq n.
\]

Using the cardinal basis, the interpolant \( I_{X,\alpha}(f)(x) \) may be written as

\[
I_{X,\alpha}(f)(x) = \sum_{j=1}^n f(x_j) \psi_j(x), \quad x \in [a, b].
\]

Some examples of cardinal bases for the EPS are plotted in Figure 2. In this illustrative example, the cardinal functions are computed for \( n = 8 \) equispaced, Halton and Chebyshev data locations, and are evaluated on 400 equispaced points.

Once the cardinal basis is computed, the Lebesgue function is defined in the usual way as

\[
\lambda(x) := \lambda(x_1, \ldots, x_n; x) := \sum_{j=1}^n |\psi_j(x)|, \quad x \in [a, b],
\]
and its maximum value is called the Lebesgue constant, defined by

\[ \Lambda := \Lambda(x_1, \ldots, x_n; x) := \sup_{a \leq x \leq b} \sum_{j=1}^{n} |\psi_j(x)|. \]

Both \( \lambda \) and \( \Lambda \) depend on the location of the interpolation points and on their number \( n \), but not on the function values and, as will be evident, they are stability indicators.

### 3.2.2 Lagrange functions and stability

It is known [23] that, in the interpolation problem, the sensitivity of the solution is determined by the condition number \( \kappa_p(\Phi) \) of the collocation matrix

\[ \kappa_p(\Phi) = \| \Phi \|_p \| \Phi^{-1} \|_p, \quad (8) \]

where \( \| \cdot \|_p \) denotes a standard operator \( p \)-norm, with \( 1 \leq p \leq \infty \). Carl de Boor [16] conjectured that the interpolation by (polynomial) B-splines of degree \( d \) at node averages is bounded by a function that depends only on \( d \), regardless of the nodes themselves. Several works disproved this conjecture; moreover improvements are also available in literature (see [21, 23, 27]). In the numerical experiments we will use (8), with \( p = 2 \), to estimate the amplification factor for the data noise.

In this work we aim to define a theoretical estimate for the residual bound, to formulate a possible stopping rule based only on the nodes distributions. At this end we relate the Lebesgue constant \( \Lambda \) and the condition number \( \kappa_2(\Phi) \).

If the interpolant is expressed in the Lagrange basis, then the Lebesgue constant also estimates the conditioning of the interpolation problem. Following [13] we define the evaluation functionals

\[ \xi_i(f) := f(x_i) = y_i, \quad i = 1, \ldots, n. \]

Then, we remark that (14 §2) the conditioning of the Lagrange representation corresponding to the evaluation functionals \( \xi_1, \ldots, \xi_n \) coincides with the Lebesgue function, i.e.

\[ \mu(x; \xi) = \sum_{i=1}^{n} |\psi_i(x)| = \lambda(x). \quad (9) \]

Moreover, the following theorem shows that the Lagrange representation, in terms of the evaluation functionals, has optimal conditioning. The conditioning of any other representation is greater than the conditioning of the Lagrange representation and the quotient can be bounded by the Skeel condition number of the inverse collocation matrix of the corresponding basis [13]. In a similar way, it also holds true for \( I_{X,a}(f) \), being expressed in cardinal form as in (7).
Theorem 2. Let \( x \in [a, b], \) \( I_{X,a}(f)(x) = \sum_{i=1}^{n} c_i \phi_i(x) \) be a representation of the Lagrange interpolation operator and let \( \Phi = (\phi_j(x_i)), \) \( i, j = 1, \ldots, n, \) be the collocation matrix. Then the following inequality holds true
\[
\mu(x; \xi) \leq \mu(x; \phi) \leq \mu_{\text{skew}}(\Phi^{-1}) \mu(x; \xi),
\]
where \( \mu(x; \xi) \) is defined in \( \Phi. \)

The conditioning measures the sensitivity of the representation to the error propagation. As concern this, let us consider the problem of interpolating \( \mu \Phi \) the Lagrange interpolation operator and let then the so-called rounding error can be bounded by \( \eta(f, x) = \sum_{j=1}^{n} |\psi_j(x)||f||_{\infty}, \)

then the so-called rounding error can be bounded by
\[
\varepsilon_{\text{rounding}}(x) = |I_{X,a}(\tilde{f}) - I_{X,a}(f)(x)| \leq \|\tilde{f} - f\|_{\infty} \eta(f, x).
\]
The rounding error, together with the approximation error
\[
\varepsilon_{\text{approx}}(x) = |f(x) - I_{X,a}(f)(x)|,
\]
defines the global error upper bound as
\[
\varepsilon(x) = \varepsilon_{\text{approx}}(x) + \varepsilon_{\text{rounding}}(x), \quad x \in [a, b].
\]

To define our greedy strategy, we essentially have to bound the approximation error, because differently from the rounding error, it is independent of the function values. To this aim we bound the associated Lebesgue function \( \lambda \) which depends only on the points. We refer to this approach as \( \lambda \)-greedy.

3.2.3 Lebesgue function and error estimation

We start by proving the following result.

Theorem 3 (Approximation error). Let \( f \in C[a, b] \) and let \( f^*_{X,a} \in \mathcal{E}_{X,a} \) be its best approximation in \( \mathcal{E}_{X,a} \) with respect to the norm \( \| \cdot \|_\infty. \) Then it holds that
\[
|(f - I_{X,a}(f))(x)| \leq (1 + \lambda(x)) \| f - f^*_{X,a} \|_\infty, \quad x \in [a, b]. \tag{10}
\]

Proof. Since \( I_{X,a}(g) = g \) for all \( g \in \mathcal{E}_{X,a}, \) and in particular for \( g = f^*_{X,a}, \) we have that
\[
|f(x) - I_{X,a}(f)(x)| = |f(x) - f^*_{X,a}(x) + f^*_{X,a}(x) - I_{X,a}(f)(x)| \\
\leq |f(x) - f^*_{X,a}(x)| + |f^*_{X,a}(x) - I_{X,a}(f)(x)| \\
= |f(x) - f^*_{X,a}(x)| + |I_{X,a}(f^*_{X,a})(x) - I_{X,a}(f)(x)| \\
= |f(x) - f^*_{X,a}(x)| + |I_{X,a}(f^*_{X,a} - f)(x)| \\
\leq \|f - f^*_{X,a}\|_\infty + |I_{X,a}(f^*_{X,a} - f)(x)|. \tag{11}
\]
To bound the second term we use (7) and thus:

\[ \left| I_{X,\alpha} \left( f_{X,a}^* - f \right)(x) \right| = \left| \sum_{j=1}^{n} (f_{X,a}^* - f)(x_j)\psi_j(x) \right| \]

\[ \leq \max_{1 \leq j \leq n} \left| (f_{X,a}^* - f)(x_j) \right| \sum_{j=1}^{n} |\psi_j(x)| \]

\[ \leq \| f_{X,a}^* - f \|_\infty \lambda(x). \]

Taking into account (11), we obtain

\[ |f(x) - I_{X,\alpha}(f)(x)| \leq \| f - f_{X,a} \|_\infty + \| f_{X,a}^* - f \|_\infty \lambda(x) \]

\[ = \| f - f_{X,a}^* \|_\infty (1 + \lambda(x)), \]

and this concludes the proof. \( \square \)

As an illustrative example, in the same setting of Figure 2 in Figure 3, we plot the Lebesgue functions associated to \( n = 8 \) equispaced, Halton and Chebyshev data and evaluated on 400 equispaced points. Observe that in this case the Chebyshev points seem to not provide the smallest Lebesgue constant. This is in contrast with interpolation with global polynomials, but in agreement with other approximation methods.

![Graphs of Lebesgue functions](image)

Figure 3: From left to right: Lebesgue functions computed on \( n = 8 \) equispaced, Halton and Chebyshev data, respectively.

**Remark 4 (Related results).** Observe that the error bound in (10) is analogous but not equivalent to similar statements in other methods (e.g., polynomial or kernel-based interpolation). Indeed, the splitting of the error on the right hand side is only partially separating the \( f \)-dependent and the \( f \)-independent terms, since the best approximant \( f_{X,a}^* \) is depending on the interpolation points. In other words, one may try to minimize the first term to find good, i.e., sub-optimal, interpolation points, but this may spoil the second term.
The \(\lambda\)-greedy algorithm

Given such error bound, we introduce a new \(\lambda\)-greedy scheme that is defined as follows. Given \(X\), \(F\) and \(\tau\), a fixed tolerance, the \(\lambda\)-greedy algorithm for exponential splines can be summarized Algorithm 3.

**Algorithm 3** Pseudo-code for the \(\lambda\)-greedy algorithm

1: Take an initial set of sorted data \(\tilde{X} = \{x_i\}_{i=1}^{q}\) \(\subset X; x_1 = a, x_q = b\) and \(q \geq 2\).
2: Compute \(\lambda(x)\) with the initial set \(\tilde{X}\).
3: While \(\arg\max_{x_i \in X \setminus \tilde{X}} \lambda(x) > \tau\):
   1. Define \(x^* = \arg\max_{x_i \in X \setminus \tilde{X}} \lambda(x)\).
   2. Set \(\tilde{X} = \tilde{X} \cup \{x^*\}\) and sort \(\tilde{X}\).
   3. Compute \(\lambda(x)\) with the set \(\tilde{X}\).

**Remark 5** (Computational aspects). Observe that the efficient execution of the \(\lambda\)-greedy algorithm depends on the efficient computation of \(\lambda\) and of \(I_{\tilde{X},\alpha}(f)\). Both of them can be computed rather efficiently by means of the local basis. Indeed, in this case for all \(x \in X\) one needs to locate the index \(i\) such that \(x \in [x_i, x_{i+1}]\), and then only perform local computations inside this interval.

**Remark 6.** In the \(\lambda\)-greedy selection, we fix a tolerance for the Lebesgue functions. However, we are able to prove the efficacy, i.e. the convergence of the \(\lambda\)-greedy scheme, only numerically. As an illustrative example, in Figure 4, we take \(n = 300\) equispaced nodes and we apply the \(\lambda\)-greedy scheme without any stopping rule, i.e. we extract \(\tilde{n} = 300\) nodes. This didactic example aims at understanding the behaviour of the Lebesgue constant when the number of nodes grows and how it relates with the conditioning of the problem. Precisely, from the first and second panel, we observe that the Lebesgue constant initially decreases and then it saturates coherently with the condition number of the interpolation matrix. In the last panel we further show the sparsity of the collocation matrix that increases as the number of nodes increases. This empirically explains the behaviour of the condition number and Lebesgue functions. In other words, our \(\lambda\)-greedy is effective until both the condition number and the Lebesgue constant do not saturate. Then, as an alternative stopping rule, one may look at the difference between the Lebesgue constant or the condition number at two consecutive iterations of the \(\lambda\)-greedy scheme.

## 4 Numerical experiments

In the following experiments, we test both the residual and the error-based schemes with different node distributions. Precisely, we consider equispaced data, Halton points and Chebyshev nodes. We further fix \(\alpha = 2\). Moreover, for all data sets, we take as initial set for the greedy strategy the first and last two
nodes. Tests have been carried out on a Intel(R) Core(TM) i7 CPU 4712MQ 2.13 GHz processor.

4.1 Testing the $f$-greedy
Throughout this subsection, we consider the following test function

$$f(x) = \text{atan}(55x), \quad x \in [-1, 1].$$

As far as the $f$-greedy method which makes use of exponential basis functions is concerned, we fix the tolerance $\tau = 10^{-3}$. In Figure 5, we plot the results obtained by taking 300 equispaced data, Halton points and Chebyshev nodes. The number of extracted greedy nodes are respectively $\tilde{n} = 36, 30$ and 36 that, as expected, cluster where the test function $f$ has steep gradients.

4.2 Testing the $\lambda$-greedy
One interesting feature of the $\lambda$-greedy scheme is that it is able to construct optimal a priori node sets, provided that a sufficiently large initial set of nodes is provided. To investigate the optimal data distribution for exponential splines, we take and initial set of 300 equispaced data and we apply the $\lambda$-greedy scheme with $\tau = 2$. The result is depicted in Figure 6 where we also show the same number of data (i.e. 32) computed with the greedy algorithm and a kernel basis (thin plate splines). For kernels, as already known in literature, the points tend to distribute in a uniform way. When using exponential splines, the greedy data tend to cluster close to the boundary, showing some similarities with Chebyshev nodes that are known to be optimal for the monomial basis.

As last experiment, in Figure 7, we plot the results of the $\lambda$-greedy scheme starting with 300 equispaced, Halton and Chebyshev data. In this case, we fix the tolerance as $\tau = 3$. The algorithm selects $\tilde{n} = 18, 19$ and 36 equispaced,
Figure 5: Results for the \( f \)-greedy algorithm. First column: the extracted greedy data (black dots), the true function \( f \) (blue dotted line) and the reconstructed function taking the greedy points (magenta solid line). Second column: the absolute error evaluated on 400 equispaced data. Third column: the maximum of the residuals at each iteration of the greedy scheme. The experiment is carried out for equispaced, Halton and Chebyshev nodes, first, second and third row, respectively.

Figure 6: Node distributions obtained via the residual-based greedy approach for EPS and kernels, respectively.
Halton and Chebyshev data, respectively. In all cases they cluster on the boundary. In the last column of Figure 7 we report the Lebesgue constant at each iteration of the greedy scheme. To get a feedback on the accuracy, with the reduced data, we reconstruct the function function $f(x) = x^2$. The associated absolute error is depicted in the second column of Figure 7.

Figure 7: Results for the $\lambda$-greedy algorithm. First column: the extracted greedy data (black dots), the true function $f$ (blue dotted line) and the reconstructed function taking the greedy points (magenta solid line). Second column: the absolute error evaluated on 400 equispaced data. Third column: the Lebesgue constant at each iteration of the greedy scheme. The experiment is carried out for equispaced, Halton and Chebyshev nodes, first, second and third row, respectively.
5 Conclusions and work in progress

We have investigated the use of greedy strategies for EPS interpolation. To this aim we have studied the cardinal form of the EPS interpolant and then we provided error bounds based on the Lebesgue functions. The results show that the error-based greedy points for EPS tend to cluster on the boundary of the approximation interval, despite the fact that Chebyshev points are not the optimal ones (this has been observed numerically via Figure 3).

Work in progress consists in investigating the proposed tool in applications, as in the context of Laplace transform inversion based on smoothing splines [12], as well as for interpolation/extrapolation algorithms for the inversion of the Fourier transform [24].

Acknowledgments

We thank the support the GNCS-INdAM project “Interpolazione e smoothing: aspetti teorici, computazionali e applicativi”. This research has been done within the Italian Network on Approximation (RITA) and the thematic group on “Approximation Theory and Applications” of the Italian Mathematical Union (UMI). EP acknowledges the financial contribution from the agreement ASI-INAF n.2018-16-HH.0.

References

[1] A. Bayliss and E. Turkel. Mappings and accuracy for Chebyshev pseudospectral approximations. J. Comput. Phys., 101:349–359, 1992.

[2] J. Berrut and H. Mittelmann. Lebesgue constant minimizing linear rational interpolation of continuous functions over the interval. Comput. Math. Appl., 33(6):77–86, 1997.

[3] P. Bohra, J. Campos, H. Gupta, S. Aziznejad, and M. Unser. Learning activation functions in deep (spline) neural networks. IEEE Open Journal of Signal Processing, 1:295–309, 2020.

[4] L. Bos, M. Caliari, S. De Marchi, M. Vianello, and Y. Xu. Bivariate Lagrange interpolation at the Padua points: The generating curve approach. J. Approx. Theory, 143(1):15–25, 2006.

[5] L. Bos, S. De Marchi, and K. Hormann. On the lebesgue constant of Berrut’s rational interpolant at equidistant nodes. J. Comput. Appl. Math., 236(4):504–510, 2011.

[6] L. Bos, S. De Marchi, and M. Vianello. Polynomial approximation on Lissajous curves in the d–cube. Appl. Num. Math., 116:47–56, 2017.
[7] L. Brutman. On the Lebesgue function for polynomial interpolation. *SIAM J. Numer. Anal.*, 15:694–704, 1978.

[8] L. Brutman. Lebesgue functions for polynomial interpolation – a survey. *Ann. Numer. Math.*, 4:111–127, 1997.

[9] R. Campagna, V. Bayona, and S. Cuomo. Using local phs+poly approximations for laplace transform inversion by gaver-stehfest algorithm. *Dolomites Res. Notes Approx.*, 13:55–64, 2020.

[10] R. Campagna and C. Conti. Penalized hyperbolic-polynomial splines. *Applied Mathematics Letters*, 118, 2021.

[11] R. Campagna, C. Conti, and S. Cuomo. Smoothing exponential-polynomial splines for multiexponential decay data. *Dolomites Res. Notes Approx.*, 12(1):86–100, 2019.

[12] R. Campagna, C. Conti, and S. Cuomo. Computational error bounds for laplace transform inversion based on smoothing splines. *Appl. Math. Comput.*, 383:125376, 2020.

[13] J. Carnicer, Y. Khiar, and J. Peña. Optimal stability of the Lagrange formula and conditioning of the Newton formula. *J. Approx. Theory*, pages 52–66, 2019.

[14] E. W. Cheney and W. A. Light. *A course in approximation theory*, volume 101. American Mathematical Soc., 2009.

[15] C. Conti, L. Romani, and D. Schenone. Semi-automatic spline fitting of planar curvilinear profiles in digital images using the Hough Transform. *Pattern Recogn.*, 74(C):64–76, Feb. 2018.

[16] C. de Boor. On local linear functionals which vanish at all b-splines but one. *Theory of Approximation with Applications*, pages 120–145, 1976.

[17] S. De Marchi, F. Marchetti, E. Perracchione, and D. Poggiali. Multivariable approximation at fake nodes. *Appl. Math. Comput.*, 391:125628, 2021.

[18] S. Dutta, M. W. Farthing, E. Perracchione, G. Savant, and M. Putti. A greedy non-intrusive reduced order model for shallow water equations. *J. Comput. Phys.*, 439:110378, 2021.

[19] G. E. Fasshauer and M. McCourt. *Kernel-based Approximation Methods using MATLAB*. World scientific, Singapore, 2015.

[20] B. Haasdonk and G. Santin. Greedy kernel approximation for sparse surrogate modeling. In W. Keiper, A. Milde, and S. Volkwein, editors, *Reduced-Order Modeling (ROM) for Simulation and Optimization: Powerful Algorithms as Key Enablers for Scientific Computing*, pages 21–45, Cham, 2018. Springer International Publishing.
[21] T. Lyche. A note on the condition numbers of the b-spline bases. *J. Approx. Theory*, 22(3):202–205, 1978.

[22] S. D. Marchi, R. Schaback, and H. Wendland. Near-optimal data-independent point locations for radial basis function interpolation. *Adv Comput Math*, 23:317–330, 2005.

[23] V. Novaković, S. Singer, and S. Singer. Estimates for the spectral condition number of cardinal b-spline collocation matrices. *Mathematical Communications*, 15(2):503–519, 2010.

[24] E. Perracchione, A. M. Massone, and M. Piana. Feature augmentation for the inversion of the fourier transform with limited data. *Inverse Problems*, aug 2021.

[25] G. Santin and B. Haasdonk. Convergence rate of the data-independent P-greedy algorithm in kernel-based approximation. *Dolomites Res. Notes Approx.*, 10(2):68–78, 2017.

[26] R. Schaback and H. Wendland. Adaptive greedy techniques for approximate solution of large RBF systems. *Numer. Algorithms*, 24(3):239–254, 2000.

[27] K. Scherer and A. Shadrin. New upper bound for the b-spline basis condition number: II. a proof of de boor's 2k-conjecture. *J. Approx. Theory*, 99(2):217–229, 1999.

[28] V. N. Temlyakov. Greedy approximation. *Acta Numer.*, 17:235–409, 2008.

[29] M. Unser. A representer theorem for deep neural networks. *J. Machine Learning Res.*, 20:1–30, 2019.

[30] H. Wendland. *Scattered Data Approximation*, volume 17 of *Cambridge Monographs on Applied and Computational Mathematics*. Cambridge University Press, Cambridge, 2005.

[31] D. Wirtz and B. Haasdonk. A Vectorial Kernel Orthogonal Greedy Algorithm. *Dolomites Res. Notes Approx.*, 6:83–100, 2013.