The Complexity of Automatic Algorithms Employing Continuous Linear Functionals

Fred J. Hickernell

Room E1-208, Department of Applied Mathematics, Illinois Institute of Technology, 10 W. 32nd St., Chicago, IL 60616

Lluís Antoni Jiménez Rugama

Room E1-120, Department of Applied Mathematics, Illinois Institute of Technology, 10 W. 32nd St., Chicago, IL 60616

Abstract

Keywords:

1. The Basic Problem

Let $\mathcal{H}_{\text{in}}$ be a separable Banach space of input functions with basis $\{u_i\}_{i\in\mathcal{I}}$, where $\mathcal{I}$ is a countable index set, and let the norm for this Banach space be defined as the $\ell_{p_{\text{in}}}$-norm terms of the Fourier coefficients as follows:

$$f = \sum_{i\in\mathcal{I}} \hat{f}_i u_i \in \mathcal{H}_{\text{in}}, \qquad \|f\|_{\mathcal{H}_{\text{in}}} = \big\|\big(\hat{f}_i\big)_{i\in\mathcal{I}}\big\|_{p_{\text{in}}}, \qquad 1 \le p_{\text{in}} \le \infty.$$

Here the $\hat{f}_i$ denotes the i^{th} Fourier coefficient of a function $f \in \mathcal{H}_{\text{in}}$. If $p_{\text{in}} = 2$, then $\mathcal{H}_{\text{in}}$ is a Hilbert space. Similarly, let $\mathcal{H}_{\text{out}}$ be a separable Banach space of outputs with basis $\{v_i\}_{i\in\mathcal{I}}$, whose norm may be defined as follows:

$$f = \sum_{i\in\mathcal{I}} \tilde{f}_i v_i \in \mathcal{H}_{\text{out}}, \qquad \|f\|_{\mathcal{H}_{\text{out}}} = \big\|\big(\tilde{f}_i\big)_{i\in\mathcal{I}}\big\|_{p_{\text{out}}}, \qquad 1 \le p_{\text{out}} \le \infty.$$

In this case $\tilde{f}_i$ denotes the i^{th} Fourier coefficient of a function $f \in \mathcal{H}_{\text{out}}$. Define the solution operator S by $S(u_i) = \lambda_i v_i$, and so

$$S(f) = \sum_{i\in\mathcal{I}}^{\infty} \lambda_i \hat{f}_i v_i, \qquad \forall f \in \mathcal{H}_{\text{in}}. \tag{1}$$

Here the λ_i are the bounded singular values of the operator S. A generalization of Hölder's inequality is

$$\|(a_i b_i)_{i\in\mathcal{I}}\|_r \le \|(a_i)_{i\in\mathcal{I}}\|_p \, \|(b_i)_{i\in\mathcal{I}}\|_q, \qquad 1 \le r, p \le \infty, \quad q = \frac{pr}{\max(p-r,0)}. \tag{2}$$

Thus, to ensure that the solution operator is bounded, it is assumed that

$$\|(\lambda_i)_{i\in\mathcal{I}}\|_q < \infty, \quad q = \frac{p_{\text{in}}p_{\text{out}}}{\max(p_{\text{in}} - p_{\text{out}}, 0)}. \tag{3}$$

This problem definition is rather general in some ways, e.g., allowing the inputs and outputs to lie in Banach spaces, rather than Hilbert spaces. However, a key requirement is that $S(u_i) = \lambda_i v_i$ for the solution operator of interest, which is a serious restriction on the choice of bases. Moreover, the norms of the Banach spaces cannot be arbitrarily defined, but must be defined in terms of the coefficients of the series expansions of the inputs and outputs.

For example, if $\mathcal{H}_{\text{in}} = \mathcal{H}_{\text{out}} = \mathcal{L}_2[0,1]$, and $S : f \mapsto f$ is the embedding operator, then one may choose a trigonometric polynomial basis:

$$\begin{gathered}\|f\|_{\mathcal{H}_{\text{in}}} = \|f\|_{\mathcal{H}_{\text{out}}} = \left[\int_0^1 |f(x)|^2 \,\mathrm{d}x\right]^{1/2}, \\ u_i(x) = v_i(x) = \mathrm{e}^{2\pi\sqrt{-1}ix}, \quad \lambda_i = 1, \qquad i \in \mathbb{Z}, \\ p_{\text{in}} = p_{\text{out}} = 2, \quad q = \infty.\end{gathered}$$

If S computes the derivative of a function, $S : f \mapsto f'$, then the space $\mathcal{H}_{\text{in}}$ must have a stronger inner product to ensure that the λ_i are bounded, e.g.,

$$\begin{gathered}\|f\|_{\mathcal{H}_{\text{in}}} = \left[\int_0^1 \left\{|f(x)|^2 + |f'(x)|^2\right\} \mathrm{d}x\right]^{1/2}, \quad \|f\|_{\mathcal{H}_{\text{out}}} = \left[\int_0^1 |f(x)|^2 \,\mathrm{d}x\right]^{1/2}, \\ u_i(x) = \frac{\mathrm{e}^{2\pi\sqrt{-1}ix}}{\sqrt{1+4\pi^2 i^2}}, \quad v_i(x) = \mathrm{e}^{2\pi\sqrt{-1}ix}, \quad \lambda_i = \frac{2\pi\sqrt{-1}i}{\sqrt{1+4\pi^2 i^2}}, \qquad i \in \mathbb{Z}, \\ p_{\text{in}} = p_{\text{out}} = 2, \quad q = \infty.\end{gathered}$$

Suppose that one may choose arbitrary linear functionals to obtain data, and let $i_1, i_2, \ldots$ be an ordering of the elements in $\mathcal{I}$. Then one may approximate $S(f)$ by the first n terms of its infinite series representation:

$$A_n(f) = \sum_{j=1}^{n} \lambda_{i_j} \hat{f}_{i_j} v_{i_j} \qquad \forall f \in \mathcal{H}_{\text{in}},\ n \in \mathbb{N}, \tag{4a}$$

$$\|S(f) - A_n(f)\|_{\mathcal{H}_{\text{out}}} = \left\| \left(\lambda_{i_j} \hat{f}_{i_j}\right)_{j\geq n+1} \right\|_{p_{\text{out}}} \tag{4b}$$

Again by the generalization of Hölder's inequality, (2), it follows that

$$\sup_{0\neq f\in\mathcal{H}_{\text{in}}} \frac{\|S(f) - A_n(f)\|_{\mathcal{H}_{\text{out}}}}{\|f\|_{\mathcal{H}_{\text{in}}}} = \left\| \left(\lambda_{i_j}\right)_{j\geq n+1} \right\|_q, \tag{4c}$$

where q was defined in (3). For the example above, a natural ordering would be $i_j = (-1)^{j-1}\lceil (j-1)/2 \rceil$. An automatic algorithm for approximating $S(f)$ must have a way of reliably bounding

$$\left\| \left(\lambda_{i_j} \hat{f}_{i_j}\right)_{j\geq n+1} \right\|_{p_{\text{out}}} \quad \text{or} \quad \left\| \left(\lambda_{i_j}\right)_{j\geq n+1} \right\|_{\infty}$$

in terms of function data. The next two sections describe two possible ways to do this.

2. Approximating a Weaker Norm as a Surrogate for a Stronger Norm

The first method for constructing an automatic algorithm assumes two Banach subspaces of input functions, $\mathcal{F} \subseteq \mathcal{G} \subseteq \mathcal{H}_{\text{in}}$, whose norms are defined as weighted $\ell_{p_{\text{in}}}$-norms of the series coefficients. Specifically, let $(\nu_i)_{i\in\mathcal{I}}$ and $(\omega_i)_{i\in\mathcal{I}}$ be positive sequences of weights, and

$$f = \sum_{i\in\mathcal{I}} \hat{f}_i u_i \in \mathcal{H}_{\text{in}}, \qquad \|f\|_{\mathcal{G}} = \left\| \left(\frac{\hat{f}_i}{\omega_i} \right)_{i\in\mathcal{I}} \right\|_{p_{\text{in}}}, \qquad \|f\|_{\mathcal{F}} = \left\| \left(\frac{\hat{f}_i}{\nu_i\omega_i} \right)_{i\in\mathcal{I}} \right\|_{p_{\text{in}}}.$$

If $\inf_{i\in\mathcal{I}} \omega_i = 0$, then the $\mathcal{G}$-norm is stronger than the $\mathcal{H}_{\text{in}}$-norm. If $\inf_{i\in\mathcal{I}} \nu_i = 0$, then the $\mathcal{F}$-norm is stronger than the $\mathcal{G}$-norm. For these two new norms the worst-case error can be bounded tightly as in (4c) as follows:

$$\begin{aligned} \sup_{0\neq f\in\mathcal{G}} \frac{\|S(f) - A_n(f)\|_{\mathcal{H}_{\text{out}}}}{\|f\|_{\mathcal{G}}} &= \left\| \left(\omega_{i_j}\lambda_{i_j}\right)_{j\geq n+1} \right\|_q, \\ \sup_{0\neq f\in\mathcal{F}} \frac{\|S(f) - A_n(f)\|_{\mathcal{H}_{\text{out}}}}{\|f\|_{\mathcal{F}}} &= \left\| \left(\nu_{i_j}\omega_{i_j}\lambda_{i_j}\right)_{j\geq n+1} \right\|_q, \end{aligned} \tag{5}$$

where again q is given in terms of p_{in} and p_{out} by (3).

The (weaker) $\mathcal{G}$-semi-norm of a function $f \in \mathcal{F}$ may be estimated using the function data $\langle u_{i_j}, f\rangle_{\mathcal{H}_{\text{in}}}$ that is also used to approximate the solution $S(f)$ as follows:

$$G_n(f) = \left\| \sum_{i=1}^{n} \hat{f}_{i_j} u_{i_j} \right\|_{\mathcal{G}} = \left\| \left(\frac{\hat{f}_{i_j}}{\omega_{i_j}} \right)_{j=1}^{n} \right\|_{p_{\text{in}}}. \tag{6}$$

The error of this approximation is given by the two-sided inequality

$$\begin{aligned} 0 \leq \|f\|_{\mathcal{G}}^{p_{\text{in}}} - G_n^{p_{\text{in}}}(f) &= \left\| \left(\frac{\hat{f}_{i_j}}{\omega_{i_j}} \right)_{j\geq n+1} \right\|_{p_{\text{in}}}^{p_{\text{in}}} \\ &\leq \left\| \left(\nu_{i_j}\right)_{j\geq n+1} \right\|_{\infty}^{p_{\text{in}}} \|f\|_{\mathcal{F}}^{p_{\text{in}}}, \quad 1 \leq p_{\text{in}} < \infty. \end{aligned}$$

Assuming f lies in the specified cone of functions, defined by

$$\mathcal{C}_\tau = \{f \in \mathcal{F} : \|f\|_{\mathcal{F}} \leq \tau \|f\|_{\mathcal{G}}\}, \tag{7}$$

the arguments of [**?**] lead to a two-sided bound on the $\mathcal{G}$-semi-norm f in terms of $G_n(f)$. Specifically, if $\left\| \left(\nu_{i_j}\right)_{j\geq n+1} \right\|_{\infty} < 1/\tau$, then for all $f \in \mathcal{C}_\tau$, it follows that

$$G_n(f) \leq \|f\|_{\mathcal{G}} \leq \mathfrak{C}_n G_n(f)$$

$$\mathfrak{C}_n = \begin{cases} \left[1 - \tau^{p_{\text{in}}} \left\| \left(\nu_{i_j}\right)_{j\geq n+1} \right\|_{\infty}^{p_{\text{in}}}\right]^{-1/p_{\text{in}}}, & 1 \leq p_{\text{in}} < \infty, \\ 1, & p_{\text{in}} = \infty. \end{cases}$$

The case $p_{\text{in}} = \infty$ follows from the $p_{\text{in}} < \infty$ case by taking the limit as $p_{\text{in}} \to \infty$. Alternatively, the case $p_{\text{in}} = \infty$ may be proven directly by choosing a positive δ strictly less than $1/\big(\tau \| \left(\nu_{i_j}\right)_{j \geq n+1} \|_\infty\big)$ and then noting that there exists $j^* \geq n+1$ with

$$\begin{aligned}\left\| \left(\frac{\hat{f}_{i_j}}{\omega_{i_j}} \right)_{j \geq n+1} \right\|_\infty &\leq \left| \frac{\hat{f}_{i_{j^*}}}{\omega_{i_{j^*}}} \right| (1+\delta) \\ &< \frac{1}{\tau \| \left(\nu_{i_j}\right)_{j \geq n+1} \|_\infty} \left| \frac{\hat{f}_{i_{j^*}}}{\omega_{i_{j^*}}} \right| \leq \frac{1}{\tau} \left| \frac{\hat{f}_{i_{j^*}}}{\nu_{i_{j^*}} \omega_{i_{j^*}}} \right| \\ &\leq \frac{1}{\tau} \left\| \left(\frac{\hat{f}_{i_j}}{\nu_{i_j} \omega_{i_j}} \right)_{j \geq n+1} \right\|_\infty \leq \frac{\|f\|_{\mathcal{F}}}{\tau} \leq \|f\|_{\mathcal{G}}\end{aligned}$$

This strict inequality implies the existence of some $j \leq n$ such that $|\hat{f}_{i_j}/\omega_{i_j}| = \|f\|_{\mathcal{G}}$, and so $G_n(f)$ has no error.

Combining this upper bound on $\|f\|_{\mathcal{G}}$, the cone condition, and error bound (5), implies that

$$\|S(f) - A_n(f)\|_{\mathcal{H}_{\text{out}}} \leq \tau \mathfrak{C}_n G_n(f) \left\| \left(\nu_{i_j} \omega_{i_j} \lambda_{i_j} \right)_{j \geq n+1} \right\|_\infty . \tag{8}$$

Thus, one increases n until the right hand side falls below some tolerance, ε. The drawback of this approach is that one needs to choose the weights ν_i and ω_i, which define the spaces $\mathcal{F}$ and $\mathcal{G}$.

2.1. Embedded Non-Adaptive Algorithms

According to [**?**], the upper bound in (8) can be used to define an algorithm using *Embedded Nonadaptive Algorithms*. Thus, with the same notation, we recall that the error of an algorithm A of this kind is:

$$\text{err}(A, \mathcal{F}, \mathcal{H}_{\text{out}}, S) = \min\{\delta \geq 0 : \|S(f) - A(f)\|_{\mathcal{H}_{\text{out}}} \leq \delta \|f\|_{\mathcal{F}} \ \ \forall f \in \mathcal{F}\}, \tag{9}$$

while the complexity of a problem for the set of algorithms $\mathcal{A}_{\text{non}}(\mathcal{F}, \mathcal{H}, S, \Lambda)$ becomes,

$$\begin{aligned}&\text{comp}(\varepsilon, \mathcal{A}_{\text{non}}(\mathcal{F}, \mathcal{H}, S, \Lambda)) \\ &\qquad = \inf \left\{ \text{cost}(A) : \text{err}(A, \mathcal{F}, \mathcal{H}_{\text{out}}, S) \leq \varepsilon, \ A \in \mathcal{A}_{\text{non}}(\mathcal{F}, \mathcal{H}, S, \Lambda) \right\}.\end{aligned} \tag{10}$$

Remember that in order to build our algorithm, we need to define a countable, non-negative-valued index set such that,

$$\mathcal{N} = \{n_1, n_2, \ldots\} \quad \text{with } n_k < n_{k+1}, \quad \text{satisfying } \sup_k \frac{n_{k+1}}{n_k} < \infty. \tag{11}$$

Algorithm 1. Consider the Banach spaces $\mathcal{F}$, $\mathcal{G}$, $\mathcal{H}_{\text{in}}$ and $\mathcal{H}_{\text{out}}$, the solution operator S and the error tolerance ε as described above. Fix τ a positive number satisfying $\left\| \left(\nu_{i_j}\right)_{j \geq n_1+1} \right\|_\infty < 1/\tau$. Moreover, let $\{A_n\}_{n \in \mathcal{I}}$, $A_n \in \mathcal{A}_{\text{non}}(\mathcal{G}, \mathcal{H}_{\text{out}}, S, \Lambda)$, be a sequence of algorithms as described in [**?**] and set $k = 1$. For any input function $f \in \mathcal{C}_\tau$, do the following:

Stage 1. Estimate the Bound. Compute $G_{n_k}(f)$ and $\mathfrak{C}_{n_k}$. The τ chosen guarantees that $\mathfrak{C}_{n_k} G_{n_k}(f)$ provides a reliable upper bound on $\|f\|_{\mathcal{G}}$.

Stage 2. Check for Convergence. Check whether k is large enough to satisfy the error tolerance, i.e.,

$$\tau \mathfrak{C}_{n_k} G_{n_k}(f) \left\| \left(\nu_{i_j} \omega_{i_j} \lambda_{i_j}\right)_{j \geq n_k+1} \right\|_{\infty} \leq \varepsilon \tag{12}$$

If this is true, then return $A_{n_k}(f)$ and terminate the algorithm.

Stage 3. Compute $k+1$. Otherwise, increase k by 1 and return to Stage 1.

For the sequence of algorithms defined in Algorithm 1, assuming that $\operatorname{err}(A, \{0\}, \mathcal{H}_{\text{out}}, S) = 0$, the equalities in (5) imply

$$\operatorname{err}(A_{n_k}, \mathcal{G}, \mathcal{H}_{\text{out}}, S) = \left\| \left(\omega_{i_j} \lambda_{i_j}\right)_{j \geq n_k+1} \right\|_q, \tag{13}$$

$$\operatorname{err}(A_{n_k}, \mathcal{F}, \mathcal{H}_{\text{out}}, S) = \left\| \left(\nu_{i_j} \omega_{i_j} \lambda_{i_j}\right)_{j \geq n_k+1} \right\|_q, \tag{14}$$

Theorem 1. *The sequence of algorithms A_n defined in Algorithm 1 is optimal for the problem $(\mathcal{J}, \mathcal{H}_{\text{out}}, S, \Lambda)$, i.e. ,*

$$\sup_{0<\varepsilon\leq 1} \frac{\min\{k \in \mathbb{N} : \operatorname{err}(A_n, \mathcal{J}, \mathcal{H}_{\text{out}}, S) \leq \varepsilon\}}{\operatorname{comp}(\varepsilon, \mathcal{A}_{\text{non}}(\mathcal{J}, \mathcal{H}_{\text{out}}, S, \Lambda))} < \infty, \qquad \mathcal{J} \in \{\mathcal{F}, \mathcal{G}\}$$

Proof. Not yet. For $\mathcal{F}$,

$$\sup_{0<\varepsilon\leq 1} \frac{\min\{k \in \mathbb{N} : \operatorname{err}(A_n, \mathcal{F}, \mathcal{H}_{\text{out}}, S) \leq \varepsilon\}}{\operatorname{comp}(\varepsilon, \mathcal{A}_{\text{non}}(\mathcal{F}, \mathcal{H}_{\text{out}}, S, \Lambda))} < \infty \tag{15}$$

$$= \sup_{0<\varepsilon\leq 1} \frac{\min\{k \in \mathbb{N} : \left\| \left(\nu_{i_j} \omega_{i_j} \lambda_{i_j}\right)_{j \geq n_k+1} \right\|_q \leq \varepsilon\}}{\operatorname{comp}(\varepsilon, \mathcal{A}_{\text{non}}(\mathcal{F}, \mathcal{H}_{\text{out}}, S, \Lambda))} < \infty \tag{16}$$

□

Because the cost of an algorithm generally increases with $\|f\|_{\mathcal{G}}$, in [**?**] the cost was also defined as:

$$\operatorname{cost}(A, \mathcal{N}, \varepsilon, N_{\max}, \sigma) = \sup\{\operatorname{cost}(A, f; \varepsilon, N_{\max}) : f \in \mathcal{N},\ \|f\|_{\mathcal{G}} \leq \sigma\}.$$

Theorem 2. *Let $\mathcal{F}$, $\mathcal{G}$, $\mathcal{H}_{\text{out}}$, ε, $\mathfrak{C}_n$ and τ be given as described above. Then the Algorithm 1 satisfies*

$$\sup_{0<\varepsilon\leq 1} \frac{\operatorname{cost}(A, \mathcal{C}_\tau, \varepsilon, \infty, \tau)}{\operatorname{comp}(\varepsilon, \mathcal{A}_{\text{non}}(\mathcal{J}, \mathcal{H}_{\text{out}}, S, \Lambda))} < \infty, \qquad \mathcal{J} \in \{\mathcal{F}, \mathcal{G}\}$$

Proof. Using theorem 1 and the proof of [**?**]. □

3. Assuming a Gentle Decay in the Terms of the Series

Another method for estimating the error of A_n assumes that the decay of the Fourier coefficients of f follows some general rate of decay, which need not be known precisely. Suppose again that we have some ordering of the linear functionals, as in the previous section, and let $0 = n_0 < n_1 < n_2 < \dots$ be an ordered, unbounded sequence of integers. Define the sums

$$\sigma_k(f) = \left\| \left(\lambda_{i_j} \hat{f}_{i_j} \right)_{j=n_{k-1}+1}^{n_k} \right\|_{p_{\text{out}}}, \qquad k = 1, 2, \dots \tag{17}$$

and the cone of functions in $\mathcal{H}_{\text{in}}$ as those whose Fourier coefficients decay at a given rate:

$$\mathcal{C} = \{ f \in \mathcal{H}_{\text{in}} : \sigma_{k+r}(f) \leq \gamma(r)\sigma_k(f),\ \forall k \in \mathbb{N} \}. \tag{18}$$

where the sequence $\gamma(r)$ has to verify $\lim_{n\to\infty} \gamma(n) = 0$, $\gamma(r) > 0$ and that

$$\left\| \underline{\gamma} \right\|_{p_{\text{out}}} = \left\| (\gamma(r))_{r=1}^{\infty} \right\|_{p_{\text{out}}} < \infty \tag{19}$$

Remark 1. The definition of the cone can be rewritten as

$$\mathcal{C} = \left\{ f \in \mathcal{H}_{\text{in}} : \sigma_k(f) \leq \min_{1 \leq i < k} \{ \gamma(i)\sigma_{k-i}(f) \},\ \forall k \in \mathbb{N} \right\}. \tag{20}$$

To see some examples on how these conditions might arise naturally, consider the case where $\gamma(r) = s_1 s_2^{-r}$ with the terms in the sum defining $\sigma_k(f)$ decaying *algebraically* and the n_k increasing *geometrically*:

$$C_{\text{lo}} j^{-p} \leq \left| \lambda_{i_j} \left\langle u_{i_j}, f \right\rangle_{\mathcal{H}_{\text{in}}} \right| \leq C_{\text{up}} j^{-p}, \quad p > 1,\ j \in \mathbb{N},$$
$$n_k = ab^k \quad a, b, k \in \mathbb{N},\ b \geq 2.$$

The sum of positive integers raised to a power can be interpreted as a left or right rectangle rule for approximating an integral. This leads to upper and lower bounds for the sums:

$$\sum_{j=n_{\text{lo}}}^{n_{\text{up}}} j^{-p} \geq \int_{n_{\text{lo}}}^{n_{\text{up}}+1} x^{-p}\, \mathrm{d}x = \frac{n_{\text{lo}}^{1-p} - (n_{\text{up}}+1)^{1-p}}{p-1},$$
$$\sum_{j=n_{\text{lo}}}^{n_{\text{up}}} j^{-p} \leq \int_{n_{\text{lo}}-1}^{n_{\text{up}}} x^{-p}\, \mathrm{d}x = \frac{(n_{\text{lo}}-1)^{1-p} - n_{\text{up}}^{1-p}}{p-1}.$$

These two bounds can be used to prove that f lies in the cone (18) for appropriately chosen s_1 and s_2:

$$\sigma_k(f) \geq C_{\text{lo}} \left\{ \frac{(n_{k-1}+1)^{1-pp_{\text{out}}} - (n_k+1)^{1-pp_{\text{out}}}}{pp_{\text{out}} - 1} \right\}^{1/p_{\text{out}}}$$
$$= C_{\text{lo}} \left\{ \frac{[ab^{k-1}]^{1-pp_{\text{out}}}}{pp_{\text{out}} - 1} \left[(1 + a^{-1}b^{1-k})^{1-pp_{\text{out}}} - (b + a^{-1}b^{1-k})^{1-pp_{\text{out}}} \right] \right\}^{1/p_{\text{out}}},$$

$$\sigma_k(f) \leq C_{\mathrm{up}} \left\{ \frac{n_{k-1}^{1-pp_{\mathrm{out}}} - n_k^{1-pp_{\mathrm{out}}}}{pp_{\mathrm{out}} - 1} \right\}^{1/p_{\mathrm{out}}}$$
$$= C_{\mathrm{up}} \left\{ \frac{[ab^{k-1}]^{1-pp_{\mathrm{out}}}(1 - b^{1-pp_{\mathrm{out}}})}{pp_{\mathrm{out}} - 1} \right\}^{1/p},$$

$$\frac{\sigma_{k+r}(f)}{\sigma_k(f)} \leq \frac{C_{\mathrm{up}}}{C_{\mathrm{lo}}} \left\{ \frac{(1 - b^{1-pp_{\mathrm{out}}})b^{(pp_{\mathrm{out}}-1)(-r)}}{(1 + a^{-1}b^{1-k})^{1-pp_{\mathrm{out}}} - (b + a^{-1}b^{1-k})^{1-pp_{\mathrm{out}}}} \right\}^{1/p_{\mathrm{out}}}$$
$$\leq \frac{C_{\mathrm{up}}}{C_{\mathrm{lo}}} \left\{ \frac{(1 - b^{1-pp_{\mathrm{out}}})b^{(pp_{\mathrm{out}}-1)(-r)}}{(1 + a^{-1})^{1-pp_{\mathrm{out}}} - (b + a^{-1})^{1-pp_{\mathrm{out}}}} \right\}^{1/p_{\mathrm{out}}} = s_1 s_2^{-r},$$

where

$$s_1 = \frac{C_{\mathrm{up}}}{C_{\mathrm{lo}}} \left\{ \frac{(1 - b^{1-pp_{\mathrm{out}}})}{(1 + a^{-1})^{1-pp_{\mathrm{out}}} - (b + a^{-1})^{1-pp_{\mathrm{out}}}} \right\}^{1/p_{\mathrm{out}}}, \qquad s_2 = b^{p-1/p_{\mathrm{out}}}.$$

One may also consider the case where the terms in the sum defining $\sigma_k(f)$ decay *exponentially* and the n_k increase *linearly*:

$$C_{\mathrm{lo}} p^{-j} \leq \left| \lambda_{i_j} \left\langle u_{i_j}, f \right\rangle_{\mathcal{H}_{\mathrm{in}}} \right| \leq C_{\mathrm{up}} p^{-j}, \quad p > 1,\ j \in \mathbb{N},$$
$$n_k = a + kb \quad a, b, k \in \mathbb{N}.$$

The geometric sum that now arises in a bound on the definition of $\sigma_k(f)$ takes the form

$$\sum_{j=n_{\mathrm{lo}}}^{n_{\mathrm{up}}} p^{-j} = \frac{p^{-n_{\mathrm{lo}}} - p^{-n_{\mathrm{up}}-1}}{1 - p^{-1}},$$

which implies that

$$\frac{\sigma_{k+r}(f)}{\sigma_k(f)} \leq \frac{C_{\mathrm{up}}}{C_{\mathrm{lo}}} \left\{ \frac{p^{-p_{\mathrm{out}}(n_{k+r-1}-1} - p^{-p_{\mathrm{out}}(n_{k+r}-1)}]}{C_{\mathrm{lo}}^2 [p^{-p_{\mathrm{out}}(n_{k-1}-1)} - p^{-p_{\mathrm{out}}(n_k-1)}]} \right\}^{1/p_{\mathrm{out}}}$$
$$= \frac{C_{\mathrm{up}} p^{n_{k-1}-n_{k+r-1}}}{C_{\mathrm{lo}}} \left\{ \frac{1 - p^{p_{\mathrm{out}}(n_{k+r-1}-n_{k+r})}}{1 - p^{p_{\mathrm{out}}(n_{k-1}-n_k)}]} \right\}^{1/p}$$
$$= \frac{C_{\mathrm{up}} p^{b(-r)}}{C_{\mathrm{lo}}} = s_1 s^{-r},$$

where

$$s_1 = \frac{C_{\mathrm{up}}}{C_{\mathrm{lo}}}, \qquad s_2 = p^b.$$

From the definition of the cone and (4b), one can show that

$$\begin{aligned}
\|S(f) - A_{n_k}(f)\|_{\mathcal{H}_{\text{out}}} &= \left\| \left(\lambda_{i_j} \hat{f}_{i_j} \right)_{j \geq n_k+1} \right\|_{p_{\text{out}}} \\
&= \left\{ \sum_{r=1}^{\infty} \sum_{j=n_{k+r-1}+1}^{n_{k+r}} \left| \lambda_{i_j} \tilde{f}_{i_j} \right|^{p_{\text{out}}} \right\}^{1/p_{\text{out}}} \\
&= \|(\sigma_{k+r}(f))_{r=1}^{\infty}\|_{p_{\text{out}}} \\
&\leq \|(\gamma(r)\sigma_k(f))_{r=1}^{\infty}\|_{p_{\text{out}}} \\
&= \sigma_k(f) \left\|\underline{\gamma}\right\|_{p_{\text{out}}}
\end{aligned} \tag{21}$$

Since the right hand side depends on the data, we have a data-driven error bound. Note also that $\sigma_k(f)$ decays as quickly with respect to n as the true error. Unlike the earlier method, one does not need to know the decay rate. In practice, one increases k until the right hand side is smaller than the error tolerance.

3.1. Adaptive algorithm

The approximate solution to the problem $S : \mathcal{C} \to \mathcal{H}_{\text{out}}$ depends on Given the error tolerance ε, we have to find k^* such that

$$\sigma_{k^*}(f) \leq \frac{\varepsilon}{\left\|\underline{\gamma}\right\|_{p_{\text{out}}}} < \sigma_{k^*-1}(f)$$

According to the inequality (21) above, this k^* is ensuring

$$\|S(f) - A_{n_{k^*}}(f)\|_{\mathcal{H}_{\text{out}}} \leq \varepsilon$$

Algorithm 2. Consider the set of functions $\mathcal{C}$, the error tolerance ε and the sequence $\{A_n\}_{n \in \mathcal{I}}$ as described before. The algorithm starts by setting $k = 1$.

Stage 1. Computing $\sigma(f)$. First we compute $\sigma_k(f)$.

Stage 2. Check for Convergence. Check whether k is large enough to satisfy the error tolerance, i.e.,

$$\sigma_k(f) \leq \frac{\varepsilon}{\left\|\underline{\gamma}\right\|_{p_{\text{out}}}} \tag{22}$$

If this is true, then we return $A_{n_k}(f)$ and terminate the algorithm.

Stage 3. Increasing k. Otherwise, we increase k by 1 and return to Stage 1.

By construction, the cost of this Algorithm 2 is

$$\text{cost}(A, f; \varepsilon) = n_{k^*} \tag{23}$$

3.2. Optimality of the Algorithm 2

The sequence $\mathcal{N}$ is used to gather the Fourier coefficients into groups to smooth the peaks we could encounter if we take these coefficients one by one. Flattening the peaks is a must to fit functions into the cone. Nevertheless, the solution of our algorithm is in this set $\mathcal{N}$ which means that the sequence does not have to distance the optimal number of data needed in our algorithm (defined below) more than a constant.

We define the optimal number of data needed in our algorithm as

$$N_{\text{opt}}(f,\varepsilon) = \min\left\{ n \in \mathbb{N} : \left\| \left(\lambda_{i_j} \hat{f}_{i_j}\right)_{j\geq n+1} \right\|_{p_{\text{out}}} \leq \varepsilon \right\}$$

The index set for our algorithm defined in (11) is called *optimal* for the Algorithm 2 and the problem $(\mathcal{C}, \mathcal{H}_{\text{out}}, S, \Lambda)$ if it essentially tracks the optimal number of data needed,

$$\sup_{0<\varepsilon\leq 1} \frac{\text{cost}(A,f;\varepsilon)}{N_{\text{opt}}(f,\varepsilon)} < \infty. \tag{24}$$

Theorem 3. *The index set defined in (11) is optimal for Algorithm 2.*

Proof. For a given f and ε, consider k^* depending on ε such that

$$\sigma_{k^*}(f) \leq \frac{\varepsilon}{\left\|\underline{\gamma}\right\|_{p_{\text{out}}}} < \sigma_{k^*-1}(f)$$

From the definition of $N_{\text{opt}}(f,\varepsilon)$, it is clear that $N_{\text{opt}}(f,\varepsilon) = n^* \leq n_{k^*}$.

Because $\lim_{n\to\infty} \gamma(n) = 0$, we can take $r = \min\{n : \gamma(n) \left\|\underline{\gamma}\right\|_{p_{\text{out}}} \leq 1\}$. Define also $\sup_k \frac{n_{k+1}}{n_k} = C$ according to the properties in (11).

Then, if $k^* > r+1$,

$$\varepsilon \leq \frac{\varepsilon}{\gamma(r)\left\|\underline{\gamma}\right\|_{p_{\text{out}}}} < \sigma_{k^*-1-r}(f) \leq \left\|S(f) - A_{n_{k^*-1-r}}(f)\right\|_{p_{\text{out}}}$$

$\left\|S(f) - A_{n^*}(f)\right\|_{p_{\text{out}}} \leq \varepsilon$

what means that $n_{k^*-1-r} < n^* \leq n_{k^*}$. Therefore,

$$\frac{n_{k^*}}{n^*} < \frac{n_{k^*}}{n_{k^*-1}} \times \cdots \times \frac{n_{k^*-r}}{n_{k^*-1-r}} \leq C^{r+1}$$

If $k^* \leq r+1$,

$$\frac{n_{k^*}}{n^*} \leq \frac{n_{k^*}}{n_{k^*-1}} \times \cdots \times \frac{n_2}{n_1} \times n_1 \leq n_1 C^{k^*-1}$$

Thus,

$$\sup_{0<\varepsilon\leq 1} \frac{\text{cost}(A,f;\varepsilon)}{N_{\text{opt}}(f,\varepsilon)} \leq n_1 C^{r+1}.$$

□

The cost of any algorithm $A \in \mathcal{A}(\mathcal{H}_{\text{in}}, \mathcal{H}_{\text{out}}, S, \Lambda)$ may also depend not only on a function, but on a set of functions $\mathcal{B} \subseteq \mathcal{H}_{\text{in}}$. To represent this idea one defines

$$\text{cost}(A, \mathcal{B}, \varepsilon) = \sup_{f \in \mathcal{B}} \{\text{cost}(A, f; \varepsilon)\}.$$

The next step is comparing our Algorithm 2 to the whole set of possible algorithms in $\mathcal{A}(\mathcal{C}, \mathcal{H}_{\text{out}}, S, \Lambda)$. If the conditions describing the cone $\mathcal{C}$ are well stated, then our algorithm will not have a greater cost than a constant times the best algorithm in $\mathcal{A}(\mathcal{C}, \mathcal{H}_{\text{out}}, S, \Lambda)$.

Lets define now the following set of functions,

$$\mathcal{J}_{n,\varepsilon} = \left\{ f \in \mathcal{H}_{\text{in}} : \left\| \left(\lambda_{i_j} \hat{f}_{i_j} \right)_{j \geq n+1} \right\|_{p_{\text{out}}} \leq \varepsilon \right\}$$

One can see that $\mathcal{J}_{n,\varepsilon} \subseteq \mathcal{J}_{m,\varepsilon}$ if $n \leq m$, and $\lim_{n\to\infty} \mathcal{J}_{n,\varepsilon} = \mathcal{H}_{\text{in}}$.

Theorem 4. *Given $A \in \mathcal{A}(\mathcal{J}_{n,\varepsilon}, \mathcal{H}_{\text{out}}, S, \Lambda)$, if $\|S(f) - A(f)\|_{\mathcal{H}_{\text{out}}} \leq \varepsilon \ \forall f \in \mathcal{J}_{n,\varepsilon}$, then*

$$\text{cost}(A, \mathcal{J}_{n,\varepsilon}, \varepsilon) \geq n$$

Proof. Consider the following subsets of $\mathcal{J}_{n,\varepsilon}$,

$$\mathcal{F}_n = \text{span}\{u_{i_1}(x), \ldots, u_{i_n}(x)\} = \{\hat{f}_{i_1} u_{i_1}(x) + \ldots + \hat{f}_{i_n} u_{i_n}(x),\ \hat{f}_{i_j} \in \mathbb{C}\},$$
$$\mathcal{F}_{n,A} = \{f \in \mathcal{F}_n :\ L_1(f) = \ldots = L_m(f) = 0,\ m < n\}.$$

where m is the cost that we suppose lower than n. Remark that,

$$\forall \varepsilon,\ \mathcal{F}_n \subseteq \mathcal{J}_{n,\varepsilon},$$
$$\forall f \in \mathcal{F}_{n,A},\ A(f) = A(0)$$

Therefore, there is the following contradiction

$$\begin{aligned} \sup_{f \in \mathcal{F}_{n,A}} \|S(f) - A(f)\|_{\mathcal{H}_{\text{out}}} &= \sup_{f \in \mathcal{F}_{n,A}} \|S(f) - A(0)\|_{\mathcal{H}_{\text{out}}} \\ &\geq \sup_{f \in \mathcal{F}_{n,A}} \|S(f)\|_{\mathcal{H}_{\text{out}}} - \|A(0)\|_{\mathcal{H}_{\text{out}}} \\ &> \varepsilon \end{aligned}$$

For the last last inequality we are using

$$\sup_{f \in \mathcal{F}_{n,A}} \|S(f)\|_{\mathcal{H}_{\text{out}}} \geq \lim_{\substack{\lambda_{i_l} \neq 0 \\ |\hat{f}_{i_l}| \to \infty}} \left\| \left(\lambda_{i_j} \hat{f}_{i_j} \right)_{j=1}^{n} \right\|_{\mathcal{H}_{\text{out}}}$$

□

The complexity of a problem for the set of algorithms $\mathcal{A}(\mathcal{B}, \mathcal{H}_{\text{out}}, S, \Lambda)$ is defined as the cost of the cheapest algorithm that satisfies the specified error tolerance, ε:

$$\begin{aligned}\text{comp}(\varepsilon,&\mathcal{A}(\mathcal{B}, \mathcal{H}_{\text{out}}, S, \Lambda)) \\ &= \inf_{A\in\mathcal{A}(\mathcal{B},\mathcal{H}_{\text{out}},S,\Lambda)} \left\{\text{cost}(A, \mathcal{B}, \varepsilon) : \|S(f) - A(f)\|_{\mathcal{H}_{\text{out}}} \leq \varepsilon,\ \forall f \in \mathcal{B}\right\}\end{aligned} \tag{25}$$

According to notation in equation (11), if $n \in \mathcal{N}, \forall f \in \mathcal{C}\cap\mathcal{J}_{n,\varepsilon}$, for Algorithm 2 we have that $n \geq \text{cost}(A; f, \varepsilon) \geq N_{\text{opt}}(f, \varepsilon)$. Thus, using Theorem 4 we know that

$$\text{comp}(\varepsilon, \mathcal{A}(\mathcal{C} \cap \mathcal{J}_{n,\varepsilon}, \mathcal{H}_{\text{out}}, S, \Lambda)) = n$$

Proposition 5 (for Theorem 6)**.** *The cost of our second algorithm tracks the optimal cost for all $f \in \mathcal{C}$*

$$\sup_{0<\varepsilon\leq 1} \sup_{f\in\mathcal{C}} \frac{\text{cost}(A, f; \varepsilon)}{\min\{n : f \in \mathcal{J}_{n,\varepsilon}\}} < \infty$$

Proof. First of all

$$\min\{n : f \in \mathcal{J}_{n,\varepsilon}\} = N_{\text{opt}}(f, \varepsilon)$$

Recalling the proof of Theorem 3, remark that $r = \min\{n : \gamma(n) \left\|\underline{\gamma}\right\|_{p_{\text{out}}} \leq 1\}$ does not depend on k^* which means that r does not depend on f

$$\sup_{0<\varepsilon\leq 1} \frac{\text{cost}(A, f; \varepsilon)}{N_{\text{opt}}(f, \varepsilon)} \leq n_1 C^{r+1}, \quad \forall f \in \mathcal{C}$$

□

Theorem 6. *Our Algorithm 2 is optimal for the problem $(\mathcal{C}, \mathcal{H}_{\text{out}}, S, \Lambda)$,*

$$\sup_{0<\varepsilon\leq 1} \frac{\text{cost}(A, \mathcal{C}, \varepsilon)}{\text{comp}(\varepsilon, \mathcal{A}(\mathcal{C}, \mathcal{H}_{\text{out}}, S, \Lambda))} < \infty$$

Proof. Because $\lim_{n\to\infty} \mathcal{J}_{n,\varepsilon} = \mathcal{H}_{\text{in}}$, there is m such that $\mathcal{C} \cap \mathcal{J}_{m,\varepsilon} \neq \emptyset$. Then,

$$\begin{aligned}\text{comp}(\varepsilon, \mathcal{A}(\mathcal{C}, \mathcal{H}_{\text{out}}, S, \Lambda)) &\geq \text{comp}(\varepsilon, \mathcal{A}(\mathcal{C} \cap \mathcal{J}_{m,\varepsilon}, \mathcal{H}_{\text{out}}, S, \Lambda)) \\ &\geq \inf_{f\in\mathcal{C}} \min\{j \in \mathbb{N} : f \in \mathcal{J}_{j,\varepsilon}\}\end{aligned}$$

The first inequality comes from the fact that $\mathcal{C} \supseteq \mathcal{C} \cap \mathcal{J}_{m,\varepsilon}$ and for the second one, $m \geq \inf_{f\in\mathcal{C}} \min\{j \in \mathbb{N} : f \in \mathcal{J}_{j,\varepsilon}\}$ as there exists $f \in \mathcal{C} \cap \mathcal{J}_{m,\varepsilon}$. ..... □

3.3. Lower Complexity Bounds for Algorithms 1 and 2

3.4. Examples

3.4.1. The embedding of $f(x) = \frac{(b^2-1)}{b^2+1-2b\cos(2\pi x)}$

In our example, consider $\mathcal{H}_{\text{in}} = \mathcal{H}_{\text{out}} = \mathcal{L}_2[0,1]$, and $S : f \mapsto f$ the embedding operator with:

$$\|f\|_{\mathcal{H}_{\text{in}}} = \|f\|_{\mathcal{H}_{\text{out}}} = \left[\int_0^1 |f(x)|^2 \, \mathrm{d}x\right]^{1/2},$$
$$u_i(x) = v_i(x) = \mathrm{e}^{2\pi\sqrt{-1}ix}, \quad \lambda_i = 1, \qquad i \in \mathbb{Z},$$
$$p_{\text{in}} = p_{\text{out}} = 2, \quad q = \infty.$$

For our index set $\mathcal{I} = \mathbb{Z} = \{0, 1, -1, 2, -2, \ldots\}$,

$$i_j = (-1)^j \left\lfloor \frac{j}{2} \right\rfloor, \qquad j \in \mathbb{N},$$
$$n_0 = 0, \quad n_k = 2k + 1, \qquad k \in \mathbb{N}.$$

The Fourier coefficients of this function can be easily found:

$$\begin{aligned}
f(x) &= \frac{b^2-1}{b^2+1-2b\cos(2\pi x)} \\
&= \frac{1 - b^{-2} + b^{-1}(\mathrm{e}^{-2\pi\sqrt{-1}x} - \mathrm{e}^{-2\pi\sqrt{-1}x})}{1 + b^{-2} - b^{-1}(\mathrm{e}^{2\pi\sqrt{-1}x} + \mathrm{e}^{-2\pi\sqrt{-1}x})} \\
&= \frac{(1 - b^{-1}\mathrm{e}^{-2\pi\sqrt{-1}x}) + (1 - b^{-1}\mathrm{e}^{2\pi\sqrt{-1}x})b^{-1}\mathrm{e}^{-2\pi\sqrt{-1}x}}{(1 - b^{-1}\mathrm{e}^{2\pi\sqrt{-1}x})(1 - b^{-1}\mathrm{e}^{-2\pi\sqrt{-1}x})} \\
&= \frac{1}{1 - b^{-1}\mathrm{e}^{2\pi\sqrt{-1}x}} + \frac{b^{-1}\mathrm{e}^{-2\pi\sqrt{-1}x}}{1 - b^{-1}\mathrm{e}^{-2\pi\sqrt{-1}x}} \\
&= \sum_{k\in\mathbb{N}_0} \left(b^{-1}\mathrm{e}^{2\pi\sqrt{-1}x}\right)^k + \sum_{k\in\mathbb{N}} \left(b^{-1}\mathrm{e}^{-2\pi\sqrt{-1}x}\right)^k \\
&= \sum_{k\in\mathbb{Z}} b^{-|k|}\mathrm{e}^{2\pi\sqrt{-1}kx} \Longrightarrow \hat{f}_{i_j} = b^{-|i_j|}
\end{aligned}$$

Because we are interested in seeing how far the approximation is from the real solution, the norm of the exact solution has to be found. Knowing the Fourier coefficients leads us to an easy way to calculate it:

$$\|S(f)\|_{\mathcal{H}_{\text{out}}} = \left(\sum_{j\in\mathbb{Z}} \lambda_{i_j}^2 \hat{f}_{i_j}^2\right)^{\frac{1}{2}} = \left(1 + 2\sum_{k\in\mathbb{N}} b^{-2k}\right)^{\frac{1}{2}} = \left(\frac{b^2+1}{b^2-1}\right)^{\frac{1}{2}}$$

We can also give the sums of coefficients,

$$\sigma_1(f) = \sqrt{1 + 2b^{-2}}$$
$$\sigma_k(f) = \sqrt{2}b^{-k}, \qquad k \in \mathbb{N} \setminus \{1\}.$$

Recalling the alternative definition of the cone in (20), with these sums, $f \in \mathcal{C}$ for $\gamma(r) = b^{-r}$ since

$$\sigma_k(f) \leq \min_{1 \leq i < k} \{\gamma(i)\sigma_{k-i}(f)\} = \min\left\{1\ ,\ \sqrt{1+\frac{b^2}{2}}\right\}\sigma_k(f)$$

In this case, $\left\|\underline{\gamma}\right\|_{p_{\text{out}}} = \frac{1}{\sqrt{b^2-1}}$. If we use our Algorithm 2, we have to check first whether $\sqrt{\frac{b^2+2}{b^4-b^2}} \leq \varepsilon$. If this inequality states, then $k^* = 1$. If not,

$$k^* = \min\left\{k \in \mathbb{N} \setminus \{1\} : \frac{1}{2\log(b)} \log\left(\frac{2}{(b^2-1)\varepsilon^2}\right) \leq k\right\}$$

For a numerical application, set $b = 3$ and find below a table of results for this example,

ε	k^*	n_{k^*}	error $= \|S(f)\|_{\mathcal{H}_{\text{out}}} - \|A_{n_{k^*}}(f)\|_{\mathcal{H}_{\text{out}}}$
1	1	3	$1.2492 \cdot 10^{-2}$
10^{-1}	2	5	$1.3811 \cdot 10^{-3}$
10^{-2}	4	9	$1.7041 \cdot 10^{-5}$
10^{-3}	6	13	$2.1038 \cdot 10^{-7}$
10^{-4}	8	17	$2.5973 \cdot 10^{-9}$
10^{-5}	10	21	$3.2065 \cdot 10^{-11}$
10^{-6}	12	25	$3.9591 \cdot 10^{-13}$
10^{-7}	15	31	$4.4409 \cdot 10^{-16}$

3.4.2. Derivation of the Bernoulli polynomials

The properties we will use for these particular polynomials can be found in [**?**]: derivatives, explicit Fourier coefficients, etc. To deal with this example, now $\mathcal{H}_{\text{in}} = \mathcal{H}_{\text{out}} = \mathcal{L}_2[0,1]$, and $S : f \mapsto f'$ with,

$$\|f\|_{\mathcal{H}_{\text{in}}} = \left[\int_0^1 \left\{|f(x)|^2 + |f'(x)|^2\right\} \mathrm{d}x\right]^{1/2}, \quad \|f\|_{\mathcal{H}_{\text{out}}} = \left[\int_0^1 |f(x)|^2 \ \mathrm{d}x\right]^{1/2},$$

$$u_i(x) = \frac{\mathrm{e}^{2\pi\sqrt{-1}ix}}{\sqrt{1+4\pi^2 i^2}}, \quad v_i(x) = \mathrm{e}^{2\pi\sqrt{-1}ix}, \quad \lambda_i = \frac{2\pi\sqrt{-1}i}{\sqrt{1+4\pi^2 i^2}}, \qquad i \in \mathbb{Z},$$

$$p_{\text{in}} = p_{\text{out}} = 2, \quad q = \infty.$$

Like in the previous example, the index set $\mathcal{I} = \mathbb{Z} = \{0, 1, -1, 2, -2, \ldots\}$ has the following ordering

$$i_j = (-1)^j \left\lfloor \frac{j}{2} \right\rfloor, \qquad j \in \mathbb{N},$$

$$n_0 = 0, \quad n_k = 2^k, \qquad k \in \mathbb{N}.$$

For this case,

$$B_n(x) = -\frac{n!}{(2\pi\sqrt{-1})^n} \sum_{k \in \mathbb{Z}\setminus\{0\}} \frac{\mathrm{e}^{2\pi\sqrt{-1}kx}}{k^n} \Longrightarrow \begin{cases} \widehat{B_{n0}} = 0 \\ \widehat{B_{ni_j}} = -\frac{n!\sqrt{1+4\pi^2 i_j^2}}{(2\pi\sqrt{-1}i_j)^n}, \quad i_j \in \mathbb{Z} \setminus \{0\} \end{cases}$$

Using that $B_n'(x) = nB_{n-1}(x)$ and $\int_0^1 B_n(t)B_m(t)\mathrm{d}t = (-1)^{n-1}\frac{m!n!}{(m+n)!}B_{n+m}$, the real solution comes automatically,

$$\|S(B_n(x))\|_{\mathcal{H}_{\text{out}}} = n\,\|B_{n-1}(x)\|_{\mathcal{H}_{\text{out}}} = n!\sqrt{\frac{\left|B_{2(n-1)}\right|}{[2(n-1)]!}}$$

where $B_n = B_n(0)$ are the Bernoulli numbers.

If we consider $\gamma(r) = s_1 s_2^{-r}$ and that

$$\frac{n!}{\sqrt{2}\pi^n} j^{-n} \leq \left|\lambda_{i_j}\widehat{B_n}_{i_j}\right| \leq \frac{n!2^n}{\pi^n} j^{-n}, \quad j \in \mathbb{N}\setminus\{1\} \tag{26}$$

then $C_{\text{up}} = 2^{n+\frac{1}{2}} C_{\text{lo}} = \frac{n!2^n}{\pi^n}$. Thus, as shown in the example example at the beginning of this section we can take

$$s_1 = 2\times 11^{n-\frac{1}{2}}, \quad s_2 = 2^{n-\frac{1}{2}}$$

There is still one thing to check because $j = 1$ was not taken into account in the inequality (26). However, see that $\sigma_1(f) \geq C_{\text{lo}}\left(\frac{5}{36}\right)^{n-\frac{1}{2}}$ what means that the $\gamma(r) = s_1 s_2^{-r}$ found satisfies our needs.

For a numerical application, find below the results for the case $n = 5$ and $n = 10$,

$B_5(x) = x^5 - \frac{5}{2}x^4 + \frac{5}{3}x^3 - \frac{1}{6}x$			
ε	k^*	n_{k^*}	error $= \|S(B_5(x))\|_{\mathcal{H}_{\text{out}}} - \|A_{n_{k^*}}(B_5(x))\|_{\mathcal{H}_{\text{out}}}$
1	5	32	$2.9439\cdot 10^{-11}$
10^{-1}	6	64	$2.2703\cdot 10^{-13}$
10^{-2}	7	128	$1.8735\cdot 10^{-15}$
10^{-3}	7	128	$1.8735\cdot 10^{-15}$
10^{-4}	8	256	$1.1102\cdot 10^{-16}$
10^{-5}	9	512	$9.7145\cdot 10^{-17}$

$B_{10}(x) = x^{10} - 5x^9 + \frac{15}{2}x^8 - 7x^6 + 5x^4 - \frac{3}{2}x^2 + \frac{5}{66}$			
ε	k^*	n_{k^*}	error $= \|S(B_{10}(x))\|_{\mathcal{H}_{\text{out}}} - \|A_{n_{k^*}}(B_{10}(x))\|_{\mathcal{H}_{\text{out}}}$
1	5	32	$7.7716\cdot 10^{-16}$
10^{-1}	5	32	$7.7716\cdot 10^{-16}$
10^{-2}	6	64	$7.7716\cdot 10^{-16}$
10^{-3}	6	64	$7.7716\cdot 10^{-16}$
10^{-4}	7	128	$7.7716\cdot 10^{-16}$
10^{-5}	7	128	$7.7716\cdot 10^{-16}$

Remark 2. MATLAB's machine epsilon is $2.2204\cdot 10^{-16}$.