as well as an update of the corresponding entries in C (c.p. (4.5.22)). Moreover, it is important to note, that the (online) complexity for computing the a-posteriori error estimator is the most expensive part we have to perform, as M_a^ϵ enters it quadratically (recall it only enters linearly for solving $\hat{P}^N(\mu)$). Hence, the factors achieved by the application of the MCEIM (c.p. Figure 4.3) become even more valuable.

In the next section we first describe the usual course of action in case of having non-affine parametric dependencies as well as possible drawbacks. Afterwards, we use what we have prepared so far in order to expand Algorithm 4.4.1, such that ϵ and δ, on which we have not commented yet, are selected automatically.

Finally, in Section 4.5.2 we try to exploit that for our particular application at hand the parametric dependency of $\underline{\underline{A}}(x;\mu)$ is basically five times one-dimensional rather than fully six-dimensional (c.p. end of Section 4.3.3) in order to reduce the complexity for evaluating $R_\epsilon^N(\mu)$ and $R_{\epsilon,\delta}^N(\mu)$.

4.5.1 ϵ-Adaptive Greedy

As already emphasized in the introduction, the key for the success of RBM is the availability of a-posteriori error estimators, as they allow for eliminating the uncertainty in the adjustment of N, the number of reduced-basis functions to use. Choosing N to large yields inefficiency, as we could achieve a desired approximation accuracy using less, whereas choosing N to small results in inaccurate approximations. Although this approach has been used for quite some time w.r.t. N as described above, it has not yet been used for deriving optimal approximation tolerances ϵ for the non-affine forms, what we want to make up for in this section.

Before we do so, however, we shortly discuss what is the current state of the art in our opinion and identify possible drawbacks. Whenever dealing with non-affine parametric dependencies the first step is to apply the EIM (c.p. Section 4.3.1) in order to obtain affine-approximations of the form (4.2.2) that seems to be sufficiently accurate and fix some M_a^ϵ, M_a^δ, where $M_a^\epsilon < M_a^\delta$. In order to stay in our notation, which is actually adapted for proceeding the other way around, namely prescribing some tolerance and deducing the number of affine terms to be used, afterwards, let ϵ (δ), such that (4.2.1a) holds true for using the prescribed number of terms M_a^ϵ (M_a^δ) in (4.2.2a). Moreover, to keep this explanation simple, we assume that f is affine consisting of M_f terms and M_f^ϵ and M_f^δ have been fixed to M_f. Then, for these fixed numbers of approximation terms the usual greedy, i.e. Algorithm 4.4.1, is applied, where for the a-posteriori error estimator $E_\epsilon^N(\mu)$ is replaced by $R_{\epsilon,\delta}^N(\mu)$ in (4.5.5) or (4.5.17), which is *not* an upper bound for $R_{\epsilon,0}^N(\mu)$.

Now, if M_a^ϵ is to small, from a certain point on, i.e. for some value of N, the error estimator becomes large due to the error introduced by approximating a by a^ϵ and not because of the approximation properties of X^N, which is usually called the EIM-plateau. Hence, once the error estimator approaches this plateau, the identification of the next parameter value to add to S^N, i.e. μ^* in Algorithm 4.4.1, based on the error estimator is almost entirely useless. Moreover,

if M_a^δ is chosen to small, e.g. $M_a^\delta = M_a^\epsilon + 1$ is a popular choice (c.p. Lemma 4.3.2), $R^N_{\epsilon,\delta}(\mu)$ usually underestimates the error by far, such that $R^N_{\epsilon,\delta}(\mu) \ll R^N_\epsilon(\mu)$ but at the same time $R^N_{\epsilon,0}(\mu) > R^N_\epsilon(\mu)$. Hence, as $E^N_\epsilon(\mu)$ has been replaced by $R^N_{\epsilon,\delta}(\mu)$, one even cannot detect that the error estimator already entered the EIM-plateau. The worst to happen at this point is, that the selected parameter μ^* already resides in S^N, which obviously breaks the greedy. Although this can certainly be repaired by restricting the search for μ^* to $\Xi^{\text{train}} \setminus S^N$ in line five of Algorithm 4.4.1, this is not a real solution for the general issue.

Hence, as one is well aware of these issues, M_a^ϵ is usually chosen conservatively large, such that (hopefully) $R^N_{\epsilon,0}(\mu) \ll R^N_\epsilon(\mu)$ at any stage of the greedy. If this holds true, the possible underestimation of $R^N_{\epsilon,0}(\mu)$ by $R^N_{\epsilon,\delta}(\mu)$, and with it the value of M_a^δ itself, becomes irrelevant, as its total contribution to the error estimator is negligible. The price to pay is obviously inefficiency, as one can do with less terms M_a^ϵ. Hence, to avoid this inefficiency at least in the online phase, one usually attaches a postprocessing phase, where the EIM-plateau is analyzed on the one hand, such that M_a^ϵ can be reduced to a more reasonable value, and on the other hand M_a^δ is analyzed and chosen, subsequently, such that the value of $R^N_{\epsilon,\delta}(\mu)$ "stabilizes".

Although this approach works fine in practice, it is somewhat unsatisfying, as for large values of M_a^ϵ the offline phase becomes considerably extensive. Moreover, it would be preferable to be able avoid the postprocessing phase, which itself can be quite extensive, too. This leads us back to the beginning of this section, namely the philosophy to choose M_a^ϵ as large as needed, but not unnecessarily large, and to use the a-posteriori error estimator for this adjustment. The resulting approach is presented in Algorithm 4.5.1, which we comment on next.

The first thing to note while comparing Algorithm 4.5.1 and Algorithm 4.4.1 is that the identification of μ^* is now embedded into a loop. Moreover, as line eight already suggests, the identified parameter value μ^* is only a candidate, where we assume that $identCrit^N_{\epsilon,\delta}(\mu)$ is $\Delta^N_{\epsilon,\delta}(\mu)$ (c.p. (4.5.24)) or some derived quantity. Note that the algorithm is designed for $0 < \delta < \epsilon$, but we could also state it for $0 = \delta < \epsilon$, which is not a good choice, however, as we will reason shortly.

Next, in line ten we only accept this candidate (i.e. give the "signal" that no more identification is necessary), if $E^N_{\epsilon,\delta}(\mu^*) \leq c_1 R^N_\epsilon(\mu^*)$, i.e. if $\Delta^N_{\epsilon,\delta}(\mu^*)$ is (sufficiently) effective. Sufficiently effective, as we can infer from Proposition 4.5.1 that the usual upper bound for the effectivity is increased by the factor $\frac{1+c_1}{1-c_1}$, which results in a factor of only two for our choice of c_1 (default values for the algorithm are given in parentheses in line two). At this point we remark that effectivity of $\Delta^N_{\epsilon,\delta}(\mu)$ is only enforced for the identified candidate μ^*, which is quite reasonable, as only for this parameter we want to ensure that $\|u(\mu) - \hat{u}^N(\mu)\|_X$ is large, too, hence $u(\mu^*)$ will make a meaningful contribution, if added to X^N in line five. For the remaining values in Ξ^{train} we content ourselves with rigorousity of $\Delta^N_{\epsilon,\delta}(\mu)$, as on termination of Algorithm 4.5.1 we want to guarantee something like (depending on the choice of $termCrit^N_{\epsilon,\delta}(\mu)$) $\|u(\mu) - \hat{u}^N(\mu)\|_X \leq \Delta^N_{\epsilon,\delta}(\mu) \leq tol$ for all $\mu \in \Xi^{\text{train}}$, where it does not matter if $\Delta^N_{\epsilon,\delta}(\mu)$ overestimates the error by far or not. To put this the other way around, assume there exist some $\tilde{\mu} \in \Xi^{\text{train}}$, such that $\|u(\tilde{\mu}) - \hat{u}^N(\tilde{\mu})\|_X$ is already quite small. Enforcing effectivity for this particular $\tilde{\mu}$ would require an unnecessary

Algorithm 4.5.1 (RBM, ϵ-Greedy Sample Selection)

1: Specify $\Xi^{\text{train}} \subset \mathcal{D}_{\text{ad}}$, $N^{\max} \in \mathbb{N}$, $tol > 0$ and $\mu^* \in \mathcal{D}_{\text{ad}}$.
2: Additionally specify $1 > c_1 (= \frac{1}{3})$, $1 > c_2 (= \frac{1}{5})$, $\epsilon (= .1)$ and $\delta (= \epsilon c_1 c_2)$.
3: Set $S^0 \leftarrow \emptyset$, $W^0 \leftarrow 0$ and $N \leftarrow 1$.
4: **while** true **do**
5: Update

$$S^N \leftarrow S^{N-1} \cup \{\mu^*\}, \quad X^N \leftarrow X^{N-1} \oplus \{u(\mu^*)\}$$

and all remaining offline quantities (w.r.t. N).
6: Set $isNotSafe \leftarrow$ true.
7: **while** $isNotSafe$ **do**
8: Identify the next candidate in Ξ^{train} to add, i.e.

$$\mu^* \leftarrow \arg\max_{\mu \in \Xi^{\text{train}}} identCrit^N_{\epsilon,\delta}(\mu).$$

9: Set $\delta^* \leftarrow c_1 c_2 \frac{R^N_\epsilon(\mu^*)}{1+\|\hat{u}^N(\mu^*)\|_X}$.
10: **if** $E^N_{\epsilon,\delta}(\mu^*) \leq c_1 R^N_\epsilon(\mu^*)$ **then**
11: Set $isNotSafe \leftarrow$ false.
12: **else**
13: Set $\epsilon \leftarrow \frac{\epsilon}{2}$.
14: **end if**
15: Set $\delta \leftarrow \min(\delta, \delta^*)$.
16: **end while**
17: **if** $(\arg\max_{\mu \in \Xi^{\text{train}}} termCrit^N_{\epsilon,\delta}(\mu) \leq tol)$ **or** $(N = N^{\max})$ **then**
18: **return**
19: **else**
20: $N \leftarrow N + 1$.
21: **end if**
22: **end while**

decrement in the approximation tolerance ϵ, hence an unnecessary increment in M^ϵ_a, which we wanted to avoid and which leads us to the adjustment of ϵ (and δ).

Assume that in line ten the candidate μ^* is rejected, as $\Delta^N_{\epsilon,\delta}(\mu^*)$ is not sufficiently effective. Moreover, assume for a moment that we have chosen to use $0 = \delta < \epsilon$, hence replaced $E^N_\epsilon(\mu)$ by $E^N_{\epsilon,0}(\mu)$ in (4.5.5) and (4.5.17), respectively, in order to obtain (4.5.24). In this case for the adjustment of ϵ we could determine ϵ^*, such that (c.p. (4.5.23))

$$c_1 = \frac{E^N_{\epsilon^*,0}(\mu^*)}{R^N_\epsilon(\mu^*)} = \frac{\epsilon^*(1 + \|\hat{u}^N(\mu^*)\|_X)}{R^N_\epsilon(\mu^*)},$$

hence set

$$\epsilon \leftarrow c_1 \frac{R^N_\epsilon(\mu^*)}{1 + \|\hat{u}^N(\mu^*)\|_X},$$

in line thirteen of Algorithm 4.5.1. However, the main issue with this approach is that $E^N_{\epsilon,0}(\mu)$ usually overestimates $R^N_{\epsilon,0}(\mu)$ by far, such that ϵ is decreased just for achieving $R^N_{\epsilon,0}(\mu^*) \ll$

$E^N_{\epsilon,0}(\mu^*) \leq c_1 R^N_\epsilon(\mu^*)$, hence again leads to an unnecessary decrement in the approximation tolerance ϵ. For this reason we have formulated Algorithm 4.5.1 w.r.t. $0 < \delta < \epsilon$, as rigorously bounding $R^N_{\epsilon,0}(\mu)$ by $E^N_{\epsilon,\delta}(\mu) = R^N_{\epsilon,\delta}(\mu) + E^N_{\delta,0}(\mu)$ allows for separating the approximation tolerance ϵ from the estimation tolerance δ, that is adapted in lines nine and fifteen, such that

$$\frac{E^N_{\delta,0}(\mu^*)}{R^N_\epsilon(\mu^*)} = \frac{\delta(1 + \|\hat{u}^N(\mu^*)\|_X)}{R^N_\epsilon(\mu^*)} \leq c_1 c_2.$$

This choice basically guarantees, that the maximum contribution of $E^N_{\delta,0}(\mu^*)$ to that value of $E^N_{\epsilon,\delta}(\mu^*)$ for which we would barely accept the identified candidate μ^* in line ten (i.e. $E^N_{\epsilon,\delta}(\mu^*) = c_1 R^N_\epsilon(\mu^*)$) is at most a fraction of c_2. Obviously, usually $E^N_{\delta,0}(\mu)$ again overestimates $R^N_{\delta,0}(\mu)$ by far. However, at this point this is not that serious, as the estimation tolerance δ can be decreased without decreasing the approximation tolerance ϵ at the same time. Moreover, in the final stages of Algorithm 4.5.1 we expect ϵ and δ to take values, such that M^ϵ_a is much larger than $M^\delta_a - M^\epsilon_a$, which is underlined by our numerical experiments in a moment. Finally, the approximation tolerance ϵ is adapted in line thirteen, where we just divide it by two. This rather crude adjustment stems from the fact that we do not know if $R^N_{\epsilon,\delta}(\mu^*)$ is currently over- or underestimating $R^N_{\epsilon,0}(\mu^*)$ and, all the more important, in which way a decrement of ϵ affects the value of $R^N_{\epsilon,\delta}(\mu^*)$.

To summarize, Algorithm 4.5.1 is enabled basically by two things. Firstly, the availability of rigorous and effective a-posteriori estimators that allows for detecting and avoiding the EIM-plateau and, secondly, the paradigm shift from specifying M^ϵ_a and deriving ϵ, afterwards, to specifying ϵ and deriving M^ϵ_a from this.

Before we are going to present the numerical experiments, we want to remark that in our opinion the proposed ϵ-greedy algorithm could very efficiently be combined with the *hp*-RBM introduced in [13]. The latter basically consists of two steps. In the first one, the admissible space of parameters is partitioned (*h*-refinement) w.r.t. some proximity indicator that we do not want to detail at this point. In the second one, a local reduced-order model is derived for each element of this partition by applying the usual greedy algorithm (*p*-refinement). Now, for non-affine parameter dependencies the application of the ϵ-greedy in the *p*-refinement step for the ith element of this partition could not only lead to a local reduced-order model w.r.t. the reduced-basis approximation space, say $X^{N_i}_i$, but also w.r.t. the necessary approximation tolerance of the non-affine forms, say ϵ_i, which we see as a chance in order to obtain even better adapted local reduced-order models.

Numerical Experiments[†]

For the numerical experiments presented in this section, we restrict our investigations to $\underline{\underline{\check{\alpha}}} = \underline{\underline{\check{\alpha}}}_{\mathrm{sym}}$, which is more challenging than $\underline{\underline{\check{\alpha}}} = \underline{\underline{\check{\alpha}}}_{\mathrm{diag}}$, but actually yields qualitative comparable results. Moreover, for mapping $\Omega \to \check{\Omega}(\mu)$ we use the non-affine mapping, introduced in Section 2.4, i.e. we choose $\tau_\mu = \tau^{\mathrm{naff}}_\mu$, whereas choosing $\tau_\mu = \tau^{\mathrm{aff}}_\mu$ yields again quite similar results. Finally,

[†]All numerical experiments have been performed using MATLAB® R2007a and COMSOL Multiphysics® 3.4.