SAC-GNC: SAmple Consensus for adaptive Graduated Non-Convexity (SUPPLEMENTARY MATERIALS)

Valter Piedade[1], Chitturi Sidhartha[2], José Gaspar[1], Venu Madhav Govindu[2], Pedro Miraldo[3]
[1]Instituto Superior Técnico, Lisboa, Portugal [2]Indian Institute of Science, Bengaluru, India
[3]Mitsubishi Electric Research Labs (MERL), Cambridge, MA, USA

*These supplementary materials present additional experiments and results in Appendix **A**, ablation studies on SAC-GNC in Appendix **B**, and details on some derivations related to GNC in Appendix **C**. In addition to this document, we have prepared a presentation illustrating the proposed tree-search strategy discussed in the main paper.*

A Additional Experiments and Results **1**
A.1. Pose graph optimization problem 1
A.2. 3D registration problem 1
A.3. Assess the robustness of the fixed annealing factor and final shape in vanilla GNC 2

B Ablation Studies **3**
B.1. Online search for σ 3
B.2. Stopping criteria 5
B.3. Shape parameter initialization 6
B.4. Robust cost function 6
B.5. Efficiency analysis 6

C Black-Rangarajan duality in GNC **6**

A. Additional Experiments and Results

This section provides additional experiments and results. Appendix **A.1** gives pose graph optimization (PGO) results in three SLAM benchmark datasets. Appendix **A.2** shows 3D registration results on one synthetic dataset. Finally, Appendix **A.3** provides further analysis on the annealing factor and final shape value for vanilla GNC across all 3DMatch sequences.

A.1. Pose graph optimization problem

We provide additional PGO results on three SLAM benchmark datasets: W100 from GTSAM in [6], M3500 from [2], and SPHERE2500 from [3, 4]. W100 and M3500 are 2D synthetic datasets simulating a simple and a challenging Manhattan world environment, respectively. SPHERE2500 is a 3D synthetic dataset.

In the main paper, we formulate the PGO problem for 2D data. Next, we show its similar formulation for 3D, which is used in some of the experiments in these supplementary materials. Given N global pose transformations $v_i \in \text{SE}(3), i = 1, \cdots, N$ from relative measurements $\tilde{e}_{i,j} \in \text{SE}(3)$, we can obtain residuals $r(\tilde{e}_{i,j}, v_i, v_j)$ for the PGO problem following

$$r(\tilde{e}_{i,j}, v_i, v_j) = \|\log\left(\tilde{e}_{i,j}^{-1}\, v_i^{-1}\, v_j\right)^{\vee}\|_{\Sigma}, \tag{A.1}$$

where $\log(\cdot)^{\vee}$ brings an element of SE(3) to its tangent space, and $\Sigma \in \mathbb{R}^{6\times 6}$ is a covariance matrix.

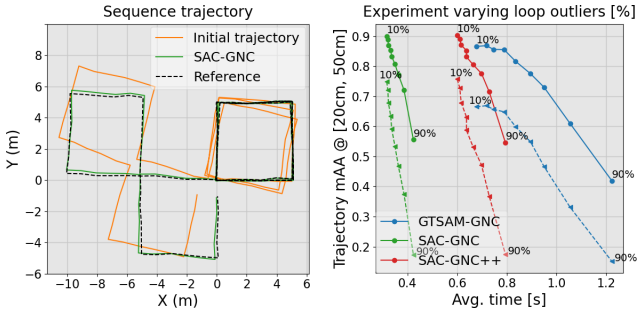
Results for the three sequences are provided in Fig. **.1**. SAC-GNC is the fastest approach, having similar or better accuracy than SAC-GNC++ and GTSAM-GNC. SAC-GNC++ also has an accuracy similar to or better than GTSAM-GNC. However, while SAC-GNC++ is faster than GTSAM-GNC in W100, it becomes slower in M3500 and SPHERE2500 due to the larger size of the graph. The mAA results for these three sequences and the two sequences shown in the main paper are detailed in Tab. **A.1**. This table clearly shows the difference in accuracy obtained starting from around 50% outliers.

A.2. 3D registration problem

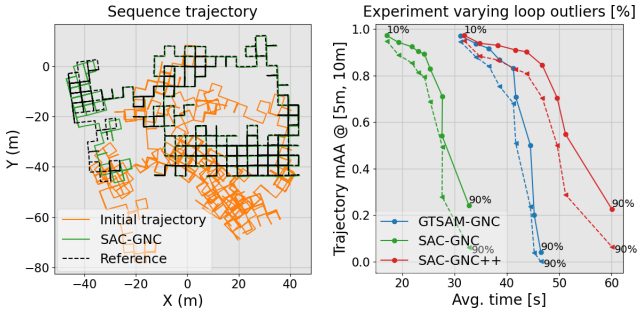
This subsection provides additional 3D registration results on synthetic data from previous GNC works, namely, the Stanford 3D scanning repository [5].

Dataset: Following GNCp's [9] data generation protocol, we generate two sets of points: 1) *Type-1*, a simple dataset with 100 points and a noise level of 0.01, and 2) *Type-2*, a more difficult dataset with $10,000$ points and noise level of 0.1. This setup was previously suggested by TEASER++ [12]. Results comparing our approach with baseline methods are shown in Tab. **A.2**.

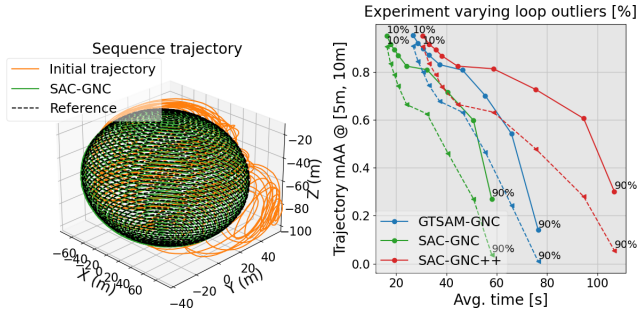
Results: We observe that on the easier *Type-1* data, SAC-GNC++ and SAC-GNC are the best and second best in rotation, while RANSAC and TEASER++ are the best and second best in translation, respectively. However, while the accuracy of SAC-GNC and SAC-GNC++ for translation is very close to RANSAC and TEASER++, SAC-GNC and SAC-GNC++ are better by a much larger margin in rotation.



(a) W100 (2D, synthetic data)



(b) M3500 (2D, synthetic data)



(c) SPHERE2500 (3D, synthetic data)

Figure .1. Pose graph optimization results on three SLAM benchmark datasets: (a) W100, (b) M3500, and (c) SPHERE2500. For each sequence, the left image displays the "Initial trajectory", the trajectory estimated by SAC-GNC with 50% outlier loops, and the "Reference" trajectory. The right image shows the trajectory mAA at 20 cm for W100 and 5 m for M3500 and SPHERE2500, represented by the dashed line, and the trajectory mAA at 50 cm for W100 and 10 m for M3500 and SPHERE2500, represented by the solid line, over the average computational time (higher mAA is better). The percentage of randomly perturbed loops varies between 10% (leftmost dot) and 90% (rightmost dot).

Also, SAC-GNC is the fastest by a large margin. On the more challenging dataset *Type-2*, SAC-GNC++ and SAC-GNC are the best and second best in rotation and translation, respectively, by a large margin when compared to RANSAC and FGR. GNCp gets fairly close to SAC-GNC and SAC-GNC++, but it is significantly slower than SAC-GNC, which remains the fastest method by a large margin.

Dataset	Method	Loop Outliers Rate								
		10%	20%	30%	40%	50%	60%	70%	80%	90%
		Trajectory mAA($e_t, 1m$)								
INTEL	GTSAM-GNC [11]	**0.963**	**0.961**	0.956	0.942	0.942	0.877	0.792	0.467	0.063
	SAC-GNC	0.954	0.954	**0.958**	**0.948**	**0.943**	0.918	**0.899**	**0.823**	0.628
	SAC-GNC++	0.955	0.955	**0.958**	0.946	0.935	**0.925**	0.896	0.818	**0.643**
		Trajectory mAA($e_t, 1m$)								
CSAIL	GTSAM-GNC [11]	0.961	0.936	**0.932**	0.909	**0.899**	0.838	0.769	0.676	0.302
	SAC-GNC	0.964	**0.946**	0.930	0.910	0.896	**0.866**	**0.802**	**0.714**	0.510
	SAC-GNC++	**0.965**	**0.946**	0.930	**0.911**	0.895	**0.866**	0.800	0.688	**0.514**
		Trajectory mAA($e_t, 50cm$)								
W100	GTSAM-GNC [11]	0.866	0.868	0.856	**0.855**	0.817	0.776	0.730	0.610	0.419
	SAC-GNC	0.900	0.889	**0.870**	0.854	**0.833**	**0.808**	0.772	**0.721**	**0.556**
	SAC-GNC++	**0.903**	**0.891**	**0.870**	0.852	0.831	0.805	**0.776**	0.715	0.546
		Trajectory mAA($e_t, 10m$)								
M3500	GTSAM-GNC [11]	0.970	0.937	0.916	0.866	0.830	0.708	0.501	0.201	0.041
	SAC-GNC	0.971	**0.942**	0.923	0.903	0.892	0.828	**0.710**	0.542	**0.242**
	SAC-GNC++	**0.972**	0.938	**0.929**	**0.909**	**0.902**	**0.844**	0.703	**0.547**	0.226
		Trajectory mAA($e_t, 10m$)								
SPHERE2500	GTSAM-GNC [11]	**0.951**	**0.919**	0.897	**0.870**	**0.829**	0.807	0.700	0.542	0.141
	SAC-GNC	0.950	0.915	**0.892**	0.869	0.823	0.809	0.714	0.598	0.270
	SAC-GNC++	0.950	0.915	**0.892**	0.866	0.823	**0.812**	**0.726**	**0.605**	**0.301**

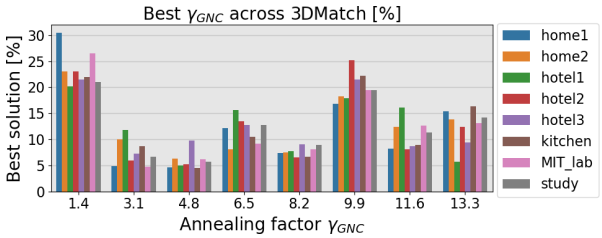
Table A.1. Additional pose graph optimization quantitative results showing trajectory mAA on multiple datasets. We highlight the **best** results.

Dataset	Method	mAA ↑				Time [ms] ↓
		$(e_R, 5°)$	$(e_R, 10°)$	$(e_t, 0.3m)$	$(e_t, 0.6m)$	
Type-1 inliers ≈ 48.8%	RANSAC [7]	0.488	0.696	**0.887**	**0.921**	1.48
	FGR [14]	0.628	0.726	0.820	0.843	1.48
	TEASER++ [12]	0.621	0.753	0.865	0.886	1.19
	GNCp [9]	0.684	0.751	0.821	0.841	3.66
	SAC-GNC	0.723	0.784	0.845	0.853	**0.93**
	SAC-GNC++	**0.736**	**0.800**	0.863	0.870	5.68
Type-2 inliers ≈ 9.8%	RANSAC [7]	0.011	0.064	0.375	0.579	168
	FGR [14]	0.619	0.726	0.796	0.828	142
	TEASER++ [12]	Runtime fail (> 30 minutes per instance)				
	GNCp [9]	0.640	0.736	0.808	0.833	157
	SAC-GNC	0.661	0.752	0.836	0.848	**102**
	SAC-GNC++	**0.666**	**0.756**	**0.839**	**0.851**	447

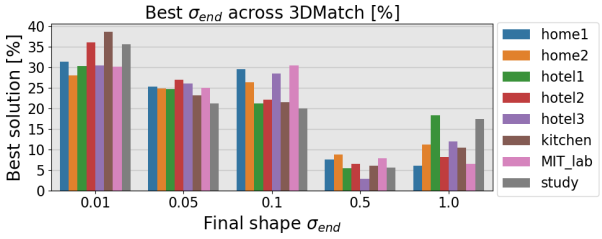
Table A.2. 3D registration results on the Stanford 3D scanning repository dataset. We highlight the **best** and second-best results.

A.3. Assess the robustness of the fixed annealing factor and final shape in vanilla GNC

To motivate the need for an adaptive annealing factor and stopping criteria that do not fully rely on a pre-defined parameter, we investigate what would be the best fixed annealing factor γ_{GNC} and final shape σ_{end} on all 3DMatch [13] dataset sequences. We start by running the vanilla GNC algorithm using arbitrary values for γ_{GNC} and $\sigma_{\text{end}} = 0.1$ to evaluate in how many sequences each value produced the best solution. We define the best solution as having the lowest rotation and translation error. Results are shown in Fig. **A.2a**. We observe that none of the tested values stand out from the rest. The annealing factors of 1.4 and 9.9 are the best performing but only get at most around 25% of the best hypothesis. Additionally, while 1.4 is the best annealing for HOME1, it is not the best for, *e.g.*, HOTEL2. As expected, this study indicates that there is no single general value for γ_{GNC} that can be predefined and give the most accurate and efficient results. We note that although 9.9 gets



(a) Study of the ideal value of γ_{GNC}



(b) Study of the ideal value of σ_{end}

Figure A.2. Study of the ideal values of (a) the annealing factor γ_{GNC} and (b) the final shape parameter σ_{end} for the vanilla GNC algorithm. We compare the percentage of best solutions acquired by each tested value for each parameter studied across different sequences of the 3DMatch dataset.

around the same amount of best solutions ($\approx 25\%$), the 9.9 mAA value is much worse overall in accuracy (rotation and translation mAA) than 1.4. This happens because the former is less robust. For $\gamma_{\text{GNC}} = 9.9$, the solutions that do not achieve the best accuracy ($\approx 75\%$ of the estimated pairs), are much worse in accuracy (rotation and translation errors) than the ones for the 1.4 case. On the other hand, for the cases in which $\gamma_{\text{GNC}} = 1.4$ losses for 9.9, it gets fairly close in terms of accuracy. That is why $\gamma_{\text{GNC}} = 1.4$ is the typically used annealing factor in the literature.

In addition, we run a similar experiment for the predefined final shape value σ_{end} for vanilla GNC. For that, we run the vanilla GNC algorithm using arbitrary values for σ_{end} and $\gamma_{\text{GNC}} = 1.4$. Results are shown in Fig. **A.2b**. We observe that each final shape value of 0.01, 0.05, or 0.1 produces around 20 to 30% of the best solutions. Again, there is no single tested value of σ_{end} that stands out from the rest as an ideal candidate.

B. Ablation Studies

This section provides ablation studies for each new technique proposed: online search for σ (Appendix **B.1**), stopping criteria (Appendix **B.2**), and σ initialization (Appendix **B.3**). The combination of these techniques constitutes the proposed SAC-GNC algorithm. Additionally, in Appendix **B.4**, we provide an ablation study testing cost functions other than the Geman-McClure. Finally, in Appendix **B.5**, we analyze the efficiency of the algorithm. All experiments use the HOME1 sequence from 3DMatch [13] in the more challenging scenario with low inlier rate matches.

B.1. Online search for σ

The search for the best shape parameter at each GNC iteration is the core of our proposed approach, and it consists of several techniques, such as the annealing selection process, model scoring, and tree search.

As written in the main paper, our experiments use $\sigma_{\text{min}} = 10^{-3}$, $\gamma_{\text{GNC}} = 1.4$, $\alpha^- = 1$, $\alpha^+ = 3.5$, our σ initialization, random annealing selection, MSAC [10] scoring, the Geman-McClure loss, and:

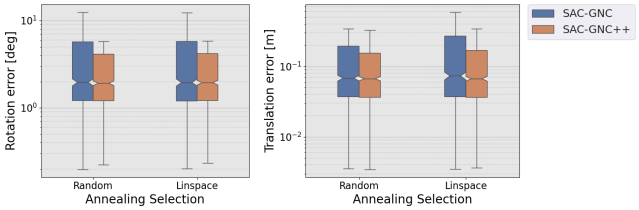
SAC-GNC : $T = 5$, $Q_{\text{add}} = 1$, and $Q_{\text{size}} = 1$;
SAC-GNC++ : $T = 10$, $Q_{\text{add}} = 2$, and $Q_{\text{size}} = 10$.

In this section, we aim to change one or two of these parameters individually and provide the results of their impact on the accuracy and efficiency of SAC-GNC.

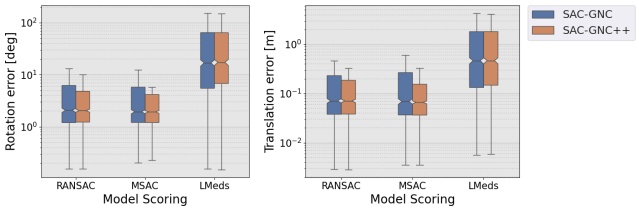
Annealing sampling: Our annealing sample approach runs several trials $t = 1, \ldots, T$ at each GNC iteration k. Each trial chooses a random annealing factor $\gamma_{k,t} \in [\gamma_{\text{GNC}} \cdot \alpha^-, \gamma_{\text{GNC}} \cdot \alpha^+]$, where $\alpha^{\pm}$ represents a relaxation of the original annealing parameter γ_{GNC}. In this paragraph, we test an alternative to random sampling. We create a linearly spaced vector of size T built with the same value constraints and sample from it (each iteration gets the annealing factor from a different vector position). Figure **B.3a** shows results comparing the accuracy of both approaches. The time difference is marginal and, therefore, not shown. We observe that results using both approaches are much alike. We opt to use the random approach as the annealing selection method for our algorithm since accuracy is similar and random annealing is more flexible. This is desirable because small variations of σ can lead to different results caught by the search-tree approach.

Model scoring: The model scoring function is crucial to our algorithm since inaccurate scores can misguide our search approach. We experiment with RANSAC [7], MSAC [10], and LMeds [8]. Figure **B.3b** shows results comparing the accuracy obtained using each method. Execution time is similar between RANSAC and MSAC, while LMeds takes slightly more time due to its sorting operation. We observe that MSAC leads to the best accuracy results, with RANSAC being a close second. LMeds' accuracy was expected to be lower since the median value can be harmful when the outlier rate exceeds 50%. However, we highlight that LMeds does not use any preset inlier threshold, which can be useful in some setups. On the other hand, we note that in most scenarios, we do have some preset idea about these inlier thresholds in practice, and their use is, therefore, justified. We choose MSAC as our model scoring function.

Annealing bounds: As described previously, the annealing selection process chooses an annealing factor $\gamma_{k,t}$ within



(a) Varying annealing selection methods



(b) Varying model scoring functions

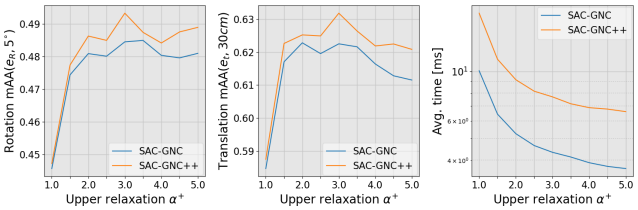
Figure B.3. Ablation studies of (a) annealing selection approaches and (b) model scoring functions.

an interval $[\gamma_{\text{GNC}} \cdot \alpha^-,\ \gamma_{\text{GNC}} \cdot \alpha^+]$. The defined interval should ensure 1) we always decrease the shape value (*i.e.*, $\gamma_{k,t} > 1$) and 2) we do not decrease the shape value too abruptly (to avoid too many jumps outside the convergence basin). We set $\alpha^- = 1$ and $\alpha^+ = 3.5$. In this section, we test the robustness of the sampler to different $\alpha^{\pm}$ parameters. Note that this is a relaxation to the typical annealing factor used in GNC-based approaches, *i.e.*, $\gamma_{\text{GNC}} = 1.4$.

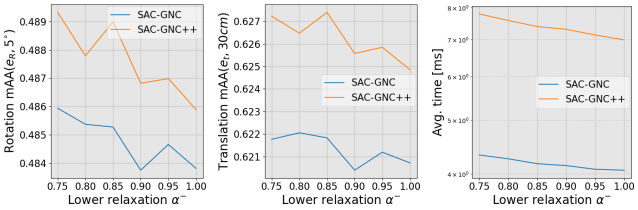
We start by inspecting the upper relaxation parameter α^+. To that end, we run an experiment varying its value with SAC-GNC and SAC-GNC++ while fixing $\alpha^- = 1$. In this context, $\alpha^+ = 1$ will correspond to vanilla GNC with a fixed annealing factor $\gamma_{\text{GNC}} = 1.4$. The results are shown in Fig. **B.4a**. We observe that, as expected, increasing α^+ decreases computational time and, eventually, accuracy. The value of α^+ leading to the best performance is 3.5.

Next, we focus on α^-. We perform a similar experiment using SAC-GNC and SAC-GNC++. This time, we fix $\alpha^+ = 3.5$ and vary α^-. Assuming $\gamma_{\text{GNC}} = 1.4$ (as done in the literature), we only vary α^- from 0.75 to 1 to ensure $1 < \gamma_{\text{GNC}} \cdot \alpha^- < \gamma_{\text{GNC}}$. The results are shown in Fig. **B.4b**. We observe that lowering α^- only leads to a slightly better accuracy (maximum improvement of approximately 0.003 in $\text{mAA}(e_R, 5°)$ and 0.002 in $\text{mAA}(e_t, 30cm)$), and slightly worse efficiency (an increase of approximately $0.3ms$ in SAC-GNC and $0.8ms$ in SAC-GNC++). These small differences indicate that changing the lower relaxation does not provide significant changes, unlike varying α^+. The best trade-off between accuracy and efficiency is obtained for $\alpha^- = 1$.

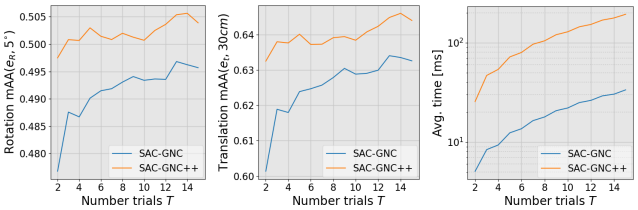
We note that the parameters $\alpha^{\pm}$ are fixed for all the experimental results in the paper, across two different problems (3D registration and PGO) and various datasets. Our



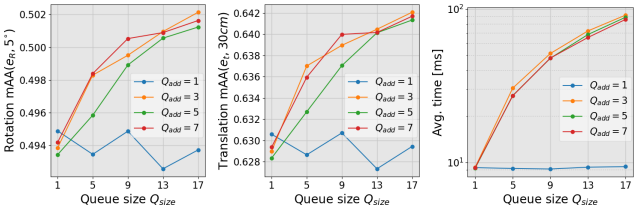
(a) Varying upper relaxation α^+ (recall the upper sampling bound is given by $\gamma_{\text{GNC}} \cdot \alpha^+$). Results using $\gamma_{\text{GNC}} = 1.4$ and $\alpha^- = 1$.



(b) Varying lower relaxation α^- (recall the lower sampling bound is given by $\gamma_{\text{GNC}} \cdot \alpha^-$). Results using $\gamma_{\text{GNC}} = 1.4$ and $\alpha^+ = 3.5$.



(c) Varying number of trials T.



(d) Varying queue size Q_{size} and maximum added hypotheses Q_{add}.

Figure B.4. Ablation studies of (a) the upper annealing relaxation α^+, (b) the lower annealing relaxation α^-,(c) the number of trials T, and (d) the maximum size of the priority queue (Q_{size}) in combination with the maximum number of hypotheses added to the queue (Q_{add}) at each iteration.

consistent improvements in all experiments demonstrate that the choice of $\alpha^{\pm}$ is robust to different unseen 3D registration and PGO data.

Number of trials: The number of trials T allows our algorithm to try various annealing factors at each GNC iteration. To test how T affects the performance of our algorithm, we vary its value from 2 to 15 using both SAC-GNC and SAC-GNC++. Results are shown in Fig. **B.4c**. As expected, increasing T improves accuracy and reduces efficiency because we are evaluating more annealing factors. For SAC-GNC specifically, we note a large jump in accuracy between 2 to 5 trials and a steady, smaller growth in accuracy as T increases beyond 5. For our general settings, we choose $T = 5$ and $T = 10$ for SAC-GNC and SAC-GNC++, re-

spectively. The aim is for SAC-GNC to strive for efficiency, while SAC-GNC++ seeks better accuracy (with a moderate efficiency loss). Note that increasing T only leads to more accurate results (by sampling more annealing factors and σ more densely), at the cost of efficiency.

Queue size and tree search rules: Our search strategy uses a priority queue to decide which shape parameter to test next. In each iteration, at most Q_{add} promising hypotheses are added to the queue. This parameter helps regulate the efficiency of the estimation. If we add all generated hypotheses, our algorithm could be computationally slow. To further improve efficiency, we define a maximum size for the priority queue Q_{size}. We run experiments using SAC-GNC and varying Q_{size} between 1 and 17, and Q_{add} between 1 and 7. Results are shown in Fig. **B.4d**. Please note that increasing these queue-related parameters can only improve accuracy, but it could come at the expense of efficiency.

We observe that, as expected, using $Q_{add} = 1$ gets similar results for all tested queue sizes. This is because only one hypothesis is added in each iteration, which is immediately used in the next. For $Q_{add} > 1$, we observe that the most critical parameter is Q_{size} since accuracy improves as Q_{size} increases and $Q_{add} = [3, 5, 7]$ produced relatively similar results. Concerning execution time, setting $Q_{add} = [3, 5, 7]$ also produced similar results. Recall that increasing Q_{add} does not mean adding Q_{add} new hypotheses each iteration. Q_{add} is an upper bound on the number of new hypotheses that can be added to the queue. This means that even though we can add more hypotheses, accuracy or time won't necessarily go up because of it. However, we note that having $Q_{add} > 1$ produces more accurate results, indicating that exploring other hypotheses is beneficial at the cost of being less efficient. For our approach, we defined the two versions (SAC-GNC and SAC-GNC++) with these results taken into account. With efficiency as the primary goal, we define SAC-GNC using $Q_{add} = 1$ and $Q_{size} = 1$. Striving for accuracy, we define SAC-GNC++ with $Q_{add} = 2$ and $Q_{size} = 5$.

Priority queue: Finally, we present an ablation study comparing two strategies for sorting the priority queue. The first strategy, **I**, prioritizes tree level first and model score second. The second strategy, **II**, considers only the model scores. Results are shown in Tab. **B.3**. We observe that **I** yields higher accuracy – especially for SAC-GNC++ – as it encourages greater exploration. In contrast, **II** results in lower accuracy and shorter runtime, as it tends to explore models predominantly from higher tree levels. This narrows the search and overlooks promising models at lower levels, particularly because scoring becomes unreliable at higher σ values (*i.e.*, it struggles to correctly distinguish inliers from outliers). Strategy **I** mitigates this issue by promoting a more balanced exploration.

Method	Priority Strategy	mAA ↑ $(e_R, 5°)$	mAA ↑ $(e_t, 0.3m)$	Iter. ↓	Time ↓ [ms]
SAC-GNC	**I**	0.489	0.622	6.73	5.54
	II	0.489	0.622	6.73	5.74
SAC-GNC++	**I**	0.505	0.643	30.4	43.4
	II	0.495	0.632	9.00	13.6

Table B.3. Ablation study on the priority queue sorting strategy: **I** sorts by tree level first, followed by model score; **II** sorts solely by model score.

Method	Stopping Criterion	σ_{min}	mAA ↑ $(e_R, 5°)$	mAA ↑ $(e_t, 0.3m)$	Iter. ↓	Time ↓ [ms]
SAC-GNC	Completed Search	10^0	0.170	0.359	1.60	1.95
		10^{-1}	0.481	0.619	3.00	3.73
		10^{-2}	0.489	0.622	5.24	5.17
		10^{-3}	0.490	0.622	7.83	6.69
	Completed Search & Convergence	10^{-3}	0.490	0.623	6.72	5.98
SAC-GNC++	Completed Search	10^0	0.231	0.427	3.81	7.72
		10^{-1}	0.501	0.638	11.2	26.0
		10^{-2}	0.506	0.645	23.8	38.4
		10^{-3}	0.507	0.645	39.8	55.7
	Completed Search & Convergence	10^{-3}	0.505	0.644	30.9	46.4

Table B.4. Ablation study on the proposed stopping criteria. Our stopping criteria let us use a $\sigma_{\text{min}} \ll \sigma_{\text{end}}$, which allows SAC-GNC to primarily converge to a solution rather than stopping at a certain σ_{end}. This improves efficiency while maintaining accuracy and enables SAC-GNC to suit diverse data.

B.2. Stopping criteria

We propose two new stopping criteria: 1) there are no more promising nodes worth exploring, *i.e.*, the queue is empty ("completed search"), and 2) the model or model scoring of the best hypothesis $\mathcal{H}^*$ converges ("convergence"). Recall that the vanilla GNC stops when a predefined σ_{end} is reached.

Table **B.4** provides results for various stopping criteria settings using SAC-GNC and SAC-GNC++. We observe that, when using only the "completed search" criterion, setting σ_{min} to 10^0 leads to the worst results, while 10^{-1}, 10^{-2}, and 10^{-3} all lead to similar results in accuracy, with efficiency lowering as σ_{min} decreases. Although for efficiency purposes, one might want to use 10^{-1} for this specific sequence, it is not obvious that it will generalize to other data. To be as general as possible, one should aim to decrease σ_{min} as much as possible[1]. However, we observe that setting σ_{min} to 10^{-3} can lead to lower efficiency. When adding the "convergence" criterion to the "completed search", we can set σ_{min} to a low value (*e.g.*, 10^{-3}). This allows the algorithm to converge to a solution rather than stopping at a certain σ_{min}, resulting in efficiency improvements for a similar accuracy. This also allows SAC-GNC to generalize

[1]Note that setting σ_{min} to a too low value can lead to null gradients when solving the optimization problem.

Method	Shape Initialization	σ_0	mAA ↑		Iter. ↓	Time ↓ [ms]
			$(e_R, 5°)$	$(e_t, 0.3m)$		
Vanilla GNC $\gamma = 1.4$	Predefined σ_0	10^3	0.465	0.612	28	7.45
	From [11]	N/A	0.458	0.609	16.3	6.08
	Ours	N/A	0.457	0.605	13.8	5.82
SAC-GNC	Predefined σ_{start}	10^3	0.487	0.618	10.9	6.80
	From [11]	N/A	0.490	0.623	8.08	6.33
	Ours	N/A	0.490	0.623	6.72	5.98
SAC-GNC++	Predefined σ_{start}	10^3	0.506	0.645	36.4	48.3
	From [11]	N/A	0.506	0.644	33.2	48.0
	Ours	N/A	0.505	0.644	30.9	46.4

Table B.5. Ablation study on the shape parameter initialization. The proposed initialization method can be used by any GNC-based approach as it depends solely on the robust cost function. N/A stands for not applicable.

Method	Cost Function $\rho_\sigma(\cdot)$	mAA ↑		Iter. ↓	Time ↓ [ms]
		$(e_R, 5°)$	$(e_t, 0.3m)$		
SAC-GNC++	Geman-McClure	0.506	0.645	36.4	48.3
	Cauchy	0.436	0.543	34.5	53.5
	Bisquare	0.359	0.533	8.76	12.7
	Logistic	0.158	0.269	21.4	25.9

Table B.6. Ablation study on robust cost functions. SAC-GNC applies to any robust loss $\rho_\sigma(\cdot)$.

to different data, not relying so heavily on hyperparameters like vanilla GNC.

B.3. Shape parameter initialization

A crucial step to avoid running unnecessary iterations is to choose a suitable σ_0 corresponding to a least squares solution for some data. Next, we test our initialization scheme against the approach of [11] and a high predefined initial σ_0 using the vanilla GNC (with $\sigma_{\text{end}} = 0.1$ and $\gamma = 1.4$), SAC-GNC, and SAC-GNC++. Results are shown in Tab. **B.5**. With vanilla GNC, we observe that our initialization leads to the lowest accuracy despite being faster. This suggests that our initialization may be too greedy, *i.e.*, it does not resemble exactly a least squares solution[2]. However, when combined with SAC-GNC or SAC-GNC++, our initialization produces similarly accurate results compared to the other initialization methods with better efficiency. This implies that, although the proposed initialization may be greedy, the search scheme of SAC-GNC can recover from it and achieve accurate and efficient solutions. The vanilla GNC is not as robust to a greedy initialization as the proposed adaptive annealing strategy.

B.4. Robust cost function

Finally, we extend the ablation study on robust cost functions provided in the main paper. Table **B.6** presents results comparing the performance of SAC-GNC++ using the Geman-McClure, Cauchy, Bisquare, and Logistic losses. Similar to the results obtained in the main paper, the Geman-McClure loss is the best-performing loss in terms of accuracy, followed by the Cauchy, Bisquare, and Logistic losses, respectively. In efficiency, it is the third-best performing. This follows the results obtained for SAC-GNC, although the efficiency drop is higher for SAC-GNC++ due to the extensive search this version of our approach conducts.

[2]Recall that, for efficiency, we define the weight of the data point with maximum residual to 0.95 and not 1 as expected in a least squares problem.

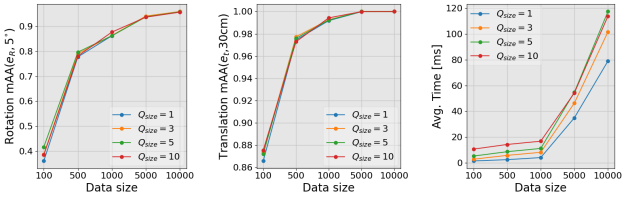


Figure B.5. Ablation study on the effect of data size M and Q_{size} on the efficiency of SAC-GNC.

B.5. Efficiency analysis

Finally, we evaluate the efficiency of SAC-GNC as a function of increasing data size M and priority queue size Q_{size}. In this ablation study, we synthetically generate data for the 3D registration problem with 40% inliers. Results for varying values of M and Q_{size} are shown in Fig. **B.5**. We observe that increasing the data size leads to reduced estimation efficiency. This outcome is expected, as the main computational bottleneck of SAC-GNC lies in the model scoring step, which has a complexity of $O(M)$. Similarly, increasing Q_{size} also results in lower efficiency, although the impact is less pronounced than that of increasing M.

Model computations: In terms of the number of models computed during estimation, SAC-GNC generates more models than the baselines, as it estimates T models per iteration – one for each sampled annealing factor. Table **B.7** compares SAC-GNC with vanilla GNC and GNCp in terms of accuracy and efficiency. As expected, SAC-GNC and SAC-GNC++ compute more models overall. However, we observe that there is no direct correlation between the number of models computed and the overall efficiency (*i.e.*, computational time). *E.g.*, comparing vanilla GNC with $\gamma = 1.4$ to GNCp (which leverages parallelism), we see that computing fewer models and performing fewer iterations does not always result in improved efficiency, due to other computational overheads.

C. Black-Rangarajan duality in GNC

To end the supplementary material, we show how to solve Eq. 2 of the paper efficiently. Although Algorithm 1 of the main paper seems simple and intuitive, solving Eq. 2 can be very costly, which is a problem since it needs to be solved several times. To alleviate this issue, GNC algorithms typically follow the Black-Rangarajan duality [1],

Dataset: 3DMatch	Annealing Update		mAA ↑		Iter. ↓	Model Comp.	Time ↓ [ms]
	Fixed γ	Adaptive	$(e_R, 5°)$	$(e_t, 0.3m)$			
↑ inliers ≈ 59.8%	1.4	–	0.641	0.809	28	28	5.82
	–	GNCp	0.636	0.801	7.80	**7.80**	4.63
	–	**SAC-GNC**	0.654	0.816	**6.40**	32	**3.85**
	–	**SAC-GNC++**	**0.655**	**0.819**	11.6	116	11.2
↓ inliers ≈ 11.6%	1.4	–	0.465	0.612	28	28	7.45
	–	GNCp	0.449	0.603	20.1	**20.1**	19.3
	–	**SAC-GNC**	0.490	0.623	**6.72**	33.6	**5.98**
	–	**SAC-GNC++**	**0.505**	**0.644**	30.9	309	46.4

Table B.7. Efficiency comparison between vanilla GNC, GNCp, and SAC-GNC. SAC-GNC performs more model computation steps than the baseline methods. However, computing fewer models does not necessarily lead to better efficiency – due to other computational overheads. We highlight the **best** and second-best results.

Algorithm C.1: *Graduated Non-Convexity*

Input – Let $\mathcal{D}$ be some data supporting the model to be estimated (including outliers and noise); σ_0 and σ_{end} be initial and final shape parameters; and γ_{GNC} be the annealing factor.

Output – Final model θ^*

Initialize: $k \leftarrow 1,\ \mathcal{W}_0 \leftarrow [1, \cdots, 1]$;
$\theta_0 \leftarrow \text{estimateModel}\,(\mathcal{D},\ \mathcal{W}_0)$;
while $\sigma_k \geq \sigma_{end}$ **do**
  $\sigma_k \leftarrow \text{updateShape}\,(\sigma_{k-1},\ \gamma_{\text{GNC}})$;
  /* SOLVE EQ. 2 OF THE PAPER */
  $\phi_0 \leftarrow \theta_{k-1}$;
  $n \leftarrow 0$;
  while *Not converged* **do**
    $n \leftarrow n + 1$;
    $\mathcal{R}_n \leftarrow \text{computeResiduals}\,(\mathcal{D},\ \phi_{n-1})$;
    $\mathcal{W}_n \leftarrow \text{computeWeights}\,(\mathcal{R}_k,\ \sigma_k)$; ▷ Eq. **C.3**
    $\phi_n \leftarrow \text{computeModel}\,(\mathcal{D},\ \mathcal{W}_k, \phi_{n-1})$; ▷ Eq. **C.6**
  $\theta_k \leftarrow \phi_n$;
  /* END */
  $k \leftarrow k + 1$;
$\theta^* \leftarrow \theta_k$;

which changes Eq. 2 into

$$\theta^*, \mathcal{W}^* = \underset{\theta, \mathcal{W}}{\text{argmin}} \sum_{i=1}^{N} \left[w_i r^2\left(\boldsymbol{x}_i, \theta\right) + \Phi_{\rho_\sigma}(w_i) \right], \quad \text{(C.2)}$$

where $\mathcal{W} \overset{\text{def}}{=} \{w_i\}$, and $w_i \in (0, 1]$ for $i = 1, \cdots, N$ are weights associated with i^{th} data sample and $\Phi_{\rho_\sigma}(w_i)$ is the outlier process. Notice that Eq. **C.2** is optimized over $\mathcal{W}$ and θ, while the original problem only optimizes over θ. However, one can see that this problem can be solved more efficiently by using alternating minimization techniques, *i.e.*, in a loop, first solving for $\mathcal{W}$ and then θ.

For the Geman-McClure robust loss function loss function in Eq. 3 of the paper, one can write (see [1])

$$w_i = \left(\frac{1}{1 + {}^{r\left(\boldsymbol{x}, \theta\right)}/_{\sigma}} \right)^2, \text{ for } i = 1, \ldots, N, \quad \text{(C.3)}$$

$$\Phi_\rho(w_i) \overset{\text{def}}{=} \sigma \left(\sqrt{w_i} - 1\right)^2. \quad \text{(C.4)}$$

Now, to solve Eq. **C.2**, again, we use an alternative minimization strategy. We start by optimizing over $\mathcal{W}$ for a fixed θ, which can be analytically solved using Eq. **C.3**. Then, we optimize Eq. **C.2** over θ for fixed $\mathcal{W}$. Since

$$\frac{\partial \Phi_\rho(w_i)}{\partial \theta} = 0, \quad \text{(C.5)}$$

solving Eq. **C.2** for θ is the same as

$$\theta^* = \underset{\theta}{\text{argmin}} \sum_{i=1}^{N} w_i r^2\left(\boldsymbol{x}_i, \theta\right), \quad \text{(C.6)}$$

which can be solved very efficiently (weighted least squares). This alternative process is represented by the inner loop in Algorithm **C.1**. As a convergence criterion, typical optimization techniques can be used. For example, we check when the distance between the estimated models θ_i and θ_{i-1} is smaller than some threshold or when the cost function value converges.

References

[1] Michael J Black and Anand Rangarajan. On the unification of line processes, outlier rejection, and robust statistics with applications in early vision. *Int. J. Comput. Vis. (IJVC)*, 19(1):57–91, 1996. 6, 7

[2] Luca Carlone, Rosario Aragues, José A Castellanos, and Basilio Bona. A fast and accurate approximation for planar pose graph optimization. *The Int. J. of Robotics Research*, 33(7):965–987, 2014. 1

[3] Luca Carlone, David M Rosen, Giuseppe Calafiore, John J Leonard, and Frank Dellaert. Lagrangian duality in 3d slam: Verification techniques and optimal solutions. In *IEEE Int. Conf. Intell. Robots Systems (IROS)*, pages 125–132, 2015. 1

[4] Luca Carlone, Roberto Tron, Kostas Daniilidis, and Frank Dellaert. Initialization techniques for 3d slam: A survey on rotation estimation and its use in pose graph optimization. In *IEEE Int. Conf. Robotics and Automation (ICRA)*, pages 4597–4604, 2015. 1

[5] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *SIGGRAPH*, pages 303–312, 1996. 1

[6] Frank Dellaert and GTSAM Contributors. borglab/gtsam. `https://github.com/borglab/gtsam`, 2022. 1

[7] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 2, 3

[8] Desire L Massart, Leonard Kaufman, Peter J Rousseeuw, and Annick Leroy. Least median of squares: a robust method for outlier and model error detection in regression and calibration. *Analytica Chimica Acta*, 187:171–179, 1986. 3

[9] Chitturi Sidhartha, Lalit Manam, and Venu Madhav Govindu. Adaptive annealing for robust geometric estimation. In *IEEE/CVF Conf. Comput. Vis. Pattern Recog. (CVPR)*, pages 21929–21939, 2023. 1, 2

[10] Philip HS Torr and Andrew Zisserman. Mlesac: A new robust estimator with application to estimating image geometry. *Comput. Vis. Image Understanding (CVIU)*, 78(1):138–156, 2000. 3

[11] Heng Yang, Pasquale Antonante, Vasileios Tzoumas, and Luca Carlone. Graduated non-convexity for robust spatial perception: From non-minimal solvers to global outlier rejection. *IEEE Robotics Automation Letters (R-AL)*, 5(2):1127–1134, 2020. 2, 6

[12] Heng Yang, Jingnan Shi, and Luca Carlone. Teaser: Fast and certifiable point cloud registration. *IEEE Trans. Robotics (T-RP)*, 37(2):314–333, 2020. 1, 2

[13] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and Thomas Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *IEEE/CVF Conf. Comput. Vis. Pattern Recog. (CVPR)*, 2017. 2, 3

[14] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Fast global registration. In *Eur. Conf. Comput. Vis. (ECCV)*, pages 766–782. Springer, 2016. 2