Trust Region Method for Noisy Functions: A Review

Anindro Bhattacharya

Abstract

Sun and Nocedal have recently proposed a trust region method to solve optimization problems involving noisy functions [1]. In this paper, we present numerical results illustrating the performance of this new algorithm compared to the naive trust region method as well as the BFGS with Errors method, an algorithm commonly used for noisy function optimization. We discovered that the new algorithm performs superior to the other two methods when there is noise only in the objective function and that all three approaches fail to converge to the unconstrained minimum when there is noise in the gradient.

1 Introduction

We consider the unconstrained minimization problem

$$\min_{x \in \mathbb{R}^n} f(x),$$

in the case that we have no access to the function $f(x)$ and/or the gradient $g(x) = \nabla f(x)$. Instead, we have access to noisy observations of $f(x)$ and $g(x)$, denoted as $\tilde{f}(x)$ and $\tilde{g}(x)$, respectively. We write

$$\tilde{f}(x) = f(x) + \delta_f(x) \text{ and } \tilde{g}(x) = g(x) + \delta_g(x),$$

where the noise $\delta_f(x)$ and $\delta_g(x)$ are assumed to be bounded by some ϵ, i.e.,

$$|\delta_f(x)| \leq \epsilon \text{ and } \|\delta_g(x)\| \leq \epsilon, \forall x \in \mathbb{R}^n.$$

One approach to solving unconstrained minimization problems is using a trust region (TR) method. TR methods define a region of radius Δ_k around the current iterate within which the trust the model sufficiently represents f and then chooses the step p_k to be the approximate minimizer of the model in this region. By comparing the step to a ratio ρ, the method decides whether the step is acceptable. If the step is not acceptable, the size of the region is reduced and a new minimizer is found [2]. One of the earliest works on TR methods is Winfield [3]. The ρ used in the Naive TR is not well suited for optimization problems where the function is noisy. To address this shortcoming, Sun and Nocedal have proposed a modified ratio [1].

One of the common algorithms currently used to solve noisy optimization problems is BFGS with Errors, proposed by Xie et al. [4]. In this paper, we compare the performance of the Noisy TR method with that of the Naive TR and BFGS with Errors methods, to highlight its strengths and underline its shortcomings.

2 Algorithms

To simplify notation in this section, f and g will be used to refer to the function and gradient, respectively, that the user has access to when running the algorithm. x_k is the solution provided by the algorithm after k iterations. Let B_k be an approximation of the Hessian on the k^{th} iteration and let H_k be an approximation of the Hessian inverse on the k^{th} iteration.

2.1 Naive Trust Region Method

In trust region methods, the step p_k at each iteration k must be determined by solving a minimization subproblem

$$\min_{p_k \in \mathbb{R}^n} m_k(p_k) \text{ s.t. } \|p_k\| \leq \Delta_k,$$

where m_k is the quadratic model

$$m_k(p_k) = f(x_k) + g(x_k)^T p_k + \frac{1}{2} p_k^T B_k p_k.$$

To decide if the step p_k should be accepted and if Δ_k should be modified, the Naive TR method uses the ratio of the actual to predicted reduction in f, defined as

$$\frac{f(x_k) - f(x_k + p_k)}{m_k(0) - m_k(p_k)}.$$

The solution is then computed using Algorithm 1.

Algorithm 1 Naive Trust Region Method (Adapted from [2])

```
1: Given $\hat{\Delta} > 0$, $\Delta_0 \in (0, \hat{\Delta})$, and $\eta \in [0, \frac{1}{4})$ :
2: for $k = 0, 1, 2, \ldots$ do
3:     Set $p_k$ to be the Cauchy point
4:     $\rho_k = \frac{f(x_k) - f(x_k + p_k)}{m_k(0) - m_k(p_k)}$
5:     if $\rho_k < \frac{1}{4}$ then
6:         $\Delta_{k+1} = \frac{1}{4}\Delta_k$
7:     else
8:         if $\rho_k > \frac{3}{4}$ and $\|p_k\| = \Delta_k$ then
9:             $\Delta_{k+1} = \min(2\Delta_k, \hat{\Delta})$
10:        else
11:            $\Delta_{k+1} = \Delta_k$
12:        end if
13:    end if
14:    if $\rho_k > \eta$ then
15:        $x_{k+1} = x_k + p_k$
16:    else
17:        $x_{k+1} = x_k$
18:    end if
19: end for
```

On Line 3, the Cauchy point is defined as $p_k^C = -\tau_k \frac{\Delta_k}{\|g_k\|} g_k$, where

$$\tau_k = \begin{cases} 1 & \text{if } g_k^T B_k g_k \leq 0 \\ \min(\|g_k\|^3/(\Delta_k g_k^T B_k g_k), 1) & \text{otherwise.} \end{cases} \tag{1}$$

2.2 Noisy Trust Region Method

In the presence of noise, the ρ_k defined in the Naive TR method is not adequate. In an attempt to resolve this, Sun and Nocedal [1] propose a modification to the reduction ratio to make it noise tolerant. The modified ratio is defined by

$$\rho_k = \frac{f(x_k) - f(x_k + p_k) + r\epsilon}{m_k(0) - m_k(p_k) + r\epsilon}$$

where $r > 2$. In the Naive TR method, a ratio close to 1 is indicative of an adequate model [2]. To remain consistent with this narrative in the presence of noise, the relaxation of both the numerator and denominator is employed by the Noisy TR method.

2.3 BFGS with Errors

BFGS is a quasi-Newton algorithm named for its discoverers Broyden, Fletcher, Goldfarb and Shanno (refer to Algorithm 6.1 in [2]). In this original BFGS method, s_k is defined to the be the displacement $x_{k+1} - x_k$ and $y_k = g(x_{k+1}) - g(x_k)$. However, when $\|s_k\| << \epsilon$, y_k can be contaminated with errors, causing the method to fail. In the BFGS with Errors method proposed in [4], the displacement s_k is lengthened by a factor l which should be at least of order $\mathcal{O}(\epsilon)$ to compensate for the error in the gradient. Algorithm 2 outlines this modified procedure.

Algorithm 2 BFGS with Errors (Adapted from [4])

```
1: Compute initial Hessian inverse approximation $H_0$
2: for $k = 0, 1, 2, \dots$ do
3:     $p_k = -H_k g(x_k)$
4:     Find stepsize $\alpha^*$ such that the Wolfe conditions are satisfied
5:     if $\alpha^*$ exists then
6:         $\alpha_k = \alpha^*$
7:     else
8:         $\alpha_k = 0$
9:     end if
10:    if $\|\alpha_k p_k\| \geq l$ then
11:        $s_k = \alpha_k p_k$
12:        $y_k = g(x_k + s_k) - g(x_k)$
13:    else
14:        $s_k = l \frac{p_k}{\|p_k\|}$
15:        $y_k = g(x_k + s_k) - g(x_k)$
16:    end if
17:    $\rho_k = \frac{1}{s_k^T y_k}$
18:    $H_{k+1} = (I - \rho_k s_k y_k^T) H_k (I - \rho_k y_k s_k^T) + \rho_k s_k s_k^T$
19:    $x_{k+1} = x_k + \alpha_k p_k$
20: end for
```

3 Numerical Experiments

We implemented the aforementioned algorithms and tested them on the function

$$\tilde{f}(x) = f(x) + \delta_f,$$

where

$$f(x) = (x_1 + 0.1, x_2 - \sin(3x_1)) A^T A (x_1 + 0.1, x_2 - \sin(3x_1))^T \text{ with } A = \begin{pmatrix} 1 & 1 \\ 1.7 & 1 \end{pmatrix}.$$

Additionally, $\delta_f \sim$ Uniform(0,1) and $\delta_g \sim$ Uniform(0,1) (i.e., $\epsilon = 1$). $f(x)$ has an unconstrained minimum at approximately [-0.1, -0.3]. Please refer to Figure 1 for the plot and contour map of the function.

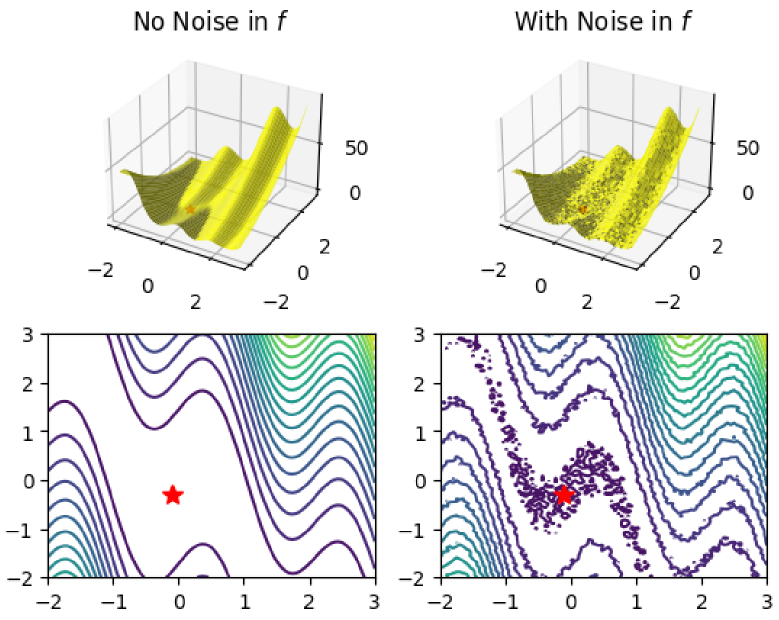

Fig. 1 Plot and Contour Map of f and Noisy f ($\tilde{f}$) where $\delta_f \sim$ Uniform(0,1). Red star marks the unconstrained minimum of the function (at approximately [-0.1, -0.3]).

We have run four sets of experiments:

- No noise in function (f) and no noise in gradient (g)
- Noise in function ($\tilde{f}$) and no noise in gradient (g)
- No noise in function (f) and noise in gradient ($\tilde{g}$)
- Noise in function ($\tilde{f}$) and noise in gradient ($\tilde{g}$)

For all experiments, the initial guess used was $x_0 = (1, 1)^T$. For trust region methods, we set $\eta = 1/6$, $\hat{\Delta} = 3$ and $\Delta_0 = 0.5$. Additionally, B_k, the approximation to the Hessian, at each iteration k was constructed using finite-differences, as follows:

$$(B_k)_{ij} = \frac{\frac{\partial}{\partial x_i} f(x_k + he_j) - \frac{\partial}{\partial x_i} f(x_k)}{h},$$

where $e_1 = [1, 0]^T$, $e_2 = [0, 1]^T$ and $h = 10^{-8}$.

For the Noisy TR method, Sun and Nocedal [1] prove that global convergence is guaranteed if $r = \frac{2}{1-c}$, where c is the constant that ρ_k is greater than in the condition on Line 8 of Algorithm 1. In this case, $c = \frac{3}{4}$. Thus, $r = 8$ was used for our numerical experiments.

For the BFGS Method with Errors, we used the Wolfe conditions constants $c_1 = 0.1$, $c_2 = 0.8$ and $\alpha_{\max} = 1$ as well as the lengthening parameter $l = 1$. We also used $H_0 = I$ as the initial Hessian inverse approximation.

For each set of experiments, the solution after 100 iterations, x_{100}, and the gradient of f at this solution, ∇f_{100}, were used to compare the performances of the algorithms. To assess the improvement of the Noisy TR method from the Naive TR method, the trends in the function values, gradient norms, ρ values and trust region radii were also analyzed.

3.1 Results

3.1.1 No noise in function (f) and no noise in gradient (g)

All three algorithms performed similarly and converged to the global minimum. Please refer to table below for x_{100} and ∇f_{100} values produced by each algorithm:

Table 1 Algorithm Performance - No noise in function (f) and no noise in gradient (g)

Algorithm	x_{100}	∇f_{100}
Naive Trust Region Method	[-0.1, -0.29552021]	[-1.11924167e-14, 5.30825384e-15]
Noisy Trust Region Method	[-0.1, -0.29552021]	[-1.11924167e-14, 5.30825384e-15]
BFGS with Errors	[-0.1, -0.29552021]	[-1.11924167e-14, 5.30825384e-15]

Although the Naive TR and Noisy TR performed similarly, the ρ values oscillated in the first 20 iterations when Naive TR was used but remained relatively consistent (around 1) when Noisy TR was used. Please refer to figure below for the trends in the function values, gradient norms, ρ values and trust region radii when TR methods were used:

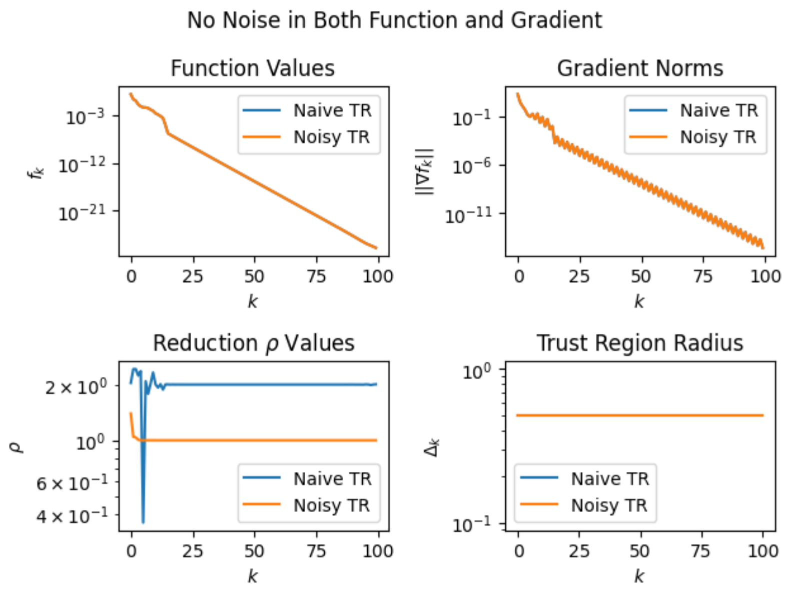

Fig. 2 Trust Region Metrics - No noise in function (f) and no noise in gradient (g)

Refer to Figures A1 and A2 in Appendix A for visualizations of the trust region radius in each iteration.

3.1.2 Noise in function ($\tilde{f}$) and no noise in gradient (g)

All three methods converge to a solution in the vicinity of the unconstrained minimum; however, there is a clear distinction in the magnitude of the gradient at the solution after 100 iterations. Having the gradient norm closer to 0 implies a better solution. With this criteria, we observed that Noisy TR performs better than BFGS with Errors by 3 magnitudes and BFGS with Errors performs better than Naive TR marginally. Please refer to table below for x_{100} and ∇f_{100} values produced by each algorithm:

Table 2 Algorithm Performance - Noise in function ($\tilde{f}$) and no noise in gradient (g)

Algorithm	x_{100}	∇f_{100}
Naive Trust Region Method	[-0.10000001, -0.29552022]	[-4.98432012e-10, -1.03373826e-09]
Noisy Trust Region Method	[-0.1, -0.29552021]	[-1.11924167e-14, 5.30825384e-15]
BFGS with Errors	[-0.10000001, -0.29552022]	[-1.9786597e-11, 9.906404e-12]

Comparing the two TR methods, we observe that ρ oscillates with Naive TR and is relatively stable with Noisy TR. Additionally, the trust region radius decreases after several iterations with Naive TR, but remains the same when Noisy TR is employed. Please refer to figure below for the trends in the function values, gradient norms, ρ values and trust region radii when TR methods were used:

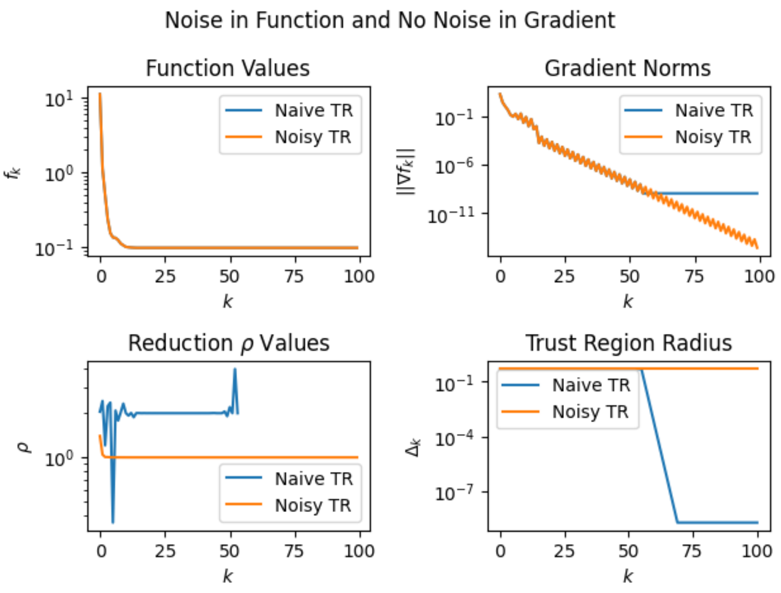

Fig. 3 Trust Region Metrics - Noise in function ($\tilde{f}$) and no noise in gradient (g)

Refer to Figures A3 and A4 in Appendix A for visualizations of the trust region radius in each iteration.

3.1.3 No noise in function (f) and noise in gradient ($\tilde{g}$)

All three methods failed to converge to a solution near the true global minimum. In terms of gradients, Naive TR and BFGS with Errors produced similar ∇f_{100}, but with Noisy TR ∇f_{100} was around machine precision. Please refer to table below for x_{100} and ∇f_{100} values produced by each algorithm:

Table 3 Algorithm Performance - No noise in function (f) and noise in gradient ($\tilde{g}$)

Algorithm	x_{100}	∇f_{100}
Naive Trust Region Method	[0.01320473, -0.09713304]	[0.0723425, 0.12980604]
Noisy Trust Region Method	[-0.39854995, -0.57622503]	[5.55111512e-17, 6.93889390e-17]
BFGS with Errors	[-0.8744164, 0.57053743]	[0.07022357, 0.13866132]

Comparing the TR methods, the function values oscillate more with Noisy TR than with Naive TR. The gradient norms quickly plateaued with Naive TR but continued to decrease for several iterations with Noisy TR. Similar to Section 3.1.2, ρ and the trust region radius is more stable with Noisy TR. Please refer to figure below for the trends in the function values, gradient norms, ρ values and trust region radii when TR methods were used:

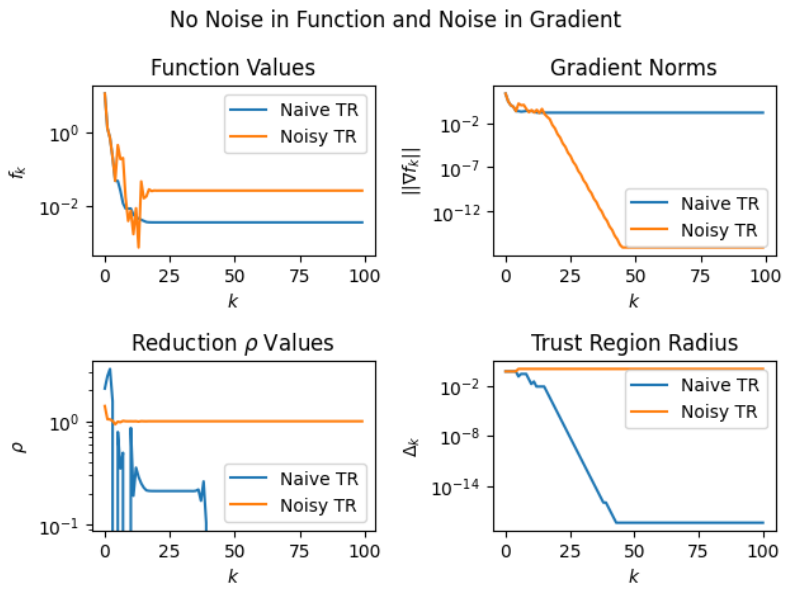

Fig. 4 Trust Region Metrics - No noise in function (f) and noise in gradient ($\tilde{g}$)

Refer to Figures A5 and A6 in Appendix A for visualizations of the trust region radius in each iteration.

3.1.4 Noise in function ($\tilde{f}$) and noise in gradient ($\tilde{g}$)

The algorithms performed similar to the observations in Section 3.1.3. Please refer to table below for x_{100} and ∇f_{100} values produced by each algorithm:

Table 4 Algorithm Performance - Noise in function ($\tilde{f}$) and noise in gradient ($\tilde{g}$)

Algorithm	x_{100}	∇f_{100}
Naive Trust Region Method	[0.01320473, -0.09713304]	[0.0723425, 0.12980604]
Noisy Trust Region Method	[-0.39854995, -0.57622503]	[5.55111512e-17, 6.93889390e-17]
BFGS with Errors	[-0.87492803, 0.57130702]	[0.06025687, 0.13615138]

Comparing the TR methods, the overall trends were relatively similar to Section 3.1.3. Please refer to figure below for the trends in the function values, gradient norms, ρ values and trust region radii when TR methods were used:

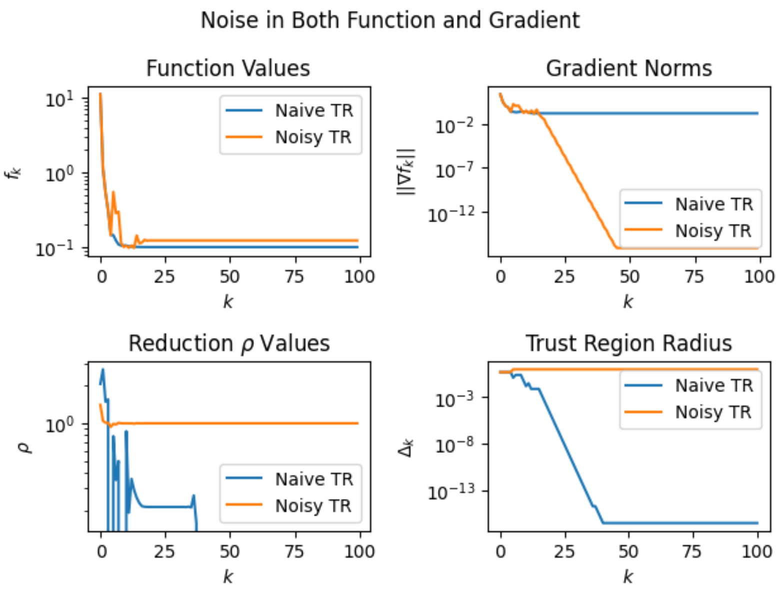

Fig. 5 Trust Region Metrics - Noise in function ($\tilde{f}$) and noise in gradient ($\tilde{g}$)

Refer to Figures A7 and A8 in Appendix A for visualizations of the trust region radius in each iteration.

3.2 Discussion

Through our numerical experiments, we made two major observations: (1) Noisy TR method outperforms Naive TR and BFGS with Errors when there is noise only in f, and (2) All methods fail when the gradient has noise. In this section, we provide potential reasons why these results were observed.

For simplicity, we will use ρ_{Naive} and ρ_{Noisy} to refer to the reduction ratio in Naive TR and Noisy TR, respectively.

3.2.1 Noisy TR method outperforms Naive TR and BFGS with Errors when there is noise only in the objective function

If Δ_k becomes very small, the numerator can be of order ϵ and the denominator will be proportional to Δ_k. So, if $\Delta_k << \epsilon$, ρ_{Naive} oscillates. These oscillations cause the trust region to shrink prematurely to the point that progress towards the minimum cannot be made. The relaxing term $r\epsilon$ in the numerator and denominator of ρ_{Noisy} prevents these oscillations resulting in a more stable trust region radius.

Although BFGS with Errors introduces a lengthening parameter to take larger steps, the calculation of the potential step size α^* may be inaccurate as we find the step size that satisfies the Wolfe conditions with respect to the noisy observation (i.e., α^* satisfies $\tilde{f}(x_k + \alpha^* p_k) \leq \tilde{f}(x_k) + c_1\alpha^* p_k^T g(x_k)$, but likely does not satisfy $f(x_k + \alpha^* p_k) \leq f(x_k) + c_1\alpha^* p_k^T g(x_k)$).

BFGS with Errors performs better than Naive TR potentially because the lengthening parameter l allows it to progress further towards the minimum compared to the prematurely shrinking trust region with Naive TR.

3.2.2 All methods fail when the gradient has noise

When computing ρ_{Naive} in Naive TR, we need to evaluate $m_k(p_k) = f(x_k) + g(x_k)^T p_k + \frac{1}{2} p_k^T B p_k$. If g has noise, the term $g(x_k)^T p_k$ and the Hessian approximation will be inaccurately computed, reducing the quality of the model substantially. This may have led to the inaccurate solution.

Since Noisy TR converges to a point with gradient around machine precision, the noisy gradient likely steers the algorithm towards a saddle point instead of the true unconstrained minimum.

When using BFGS with Errors, the noise in the gradient likely results in choosing a suboptimal step size α^* (similar to when there is noise in the function). Additionally, the computation of the Hessian inverse approximation H_k was inaccurate due to the erratice nature of the change in the gradient $y_k = g(x_k - s_k) - g(x_k)$.

4 Conclusion

As can be seen, the Noisy TR method outperforms both the Naive TR and BFGS with Error methods when there is noise present in only the function; however, all of the methods fail when there is noise in the gradient. Thus, solving optimization problems with noisy gradients should be a topic of focus in future work.

An approach to consider is adding a pre-processing step where a smoothing method can be applied to the noisy gradient. Perhaps, the behaviour of the methods will be less erratic with this smoothed gradient. Additionally, alternate approximations to B could explored. Finite difference applied to $\tilde{g}$ likely provides sub-optimal approximations to the second derivatives, especially when there is a great deal of noise. Lastly, it may not be worth the effort to use the noisy gradient altogether. Instead, using derivative-free optimization methods may provide superior results.

Appendix A Trust Region Radii Visualizations

Please refer to the figures below for the visualizations of the trust region radii over the first 100 iterations:

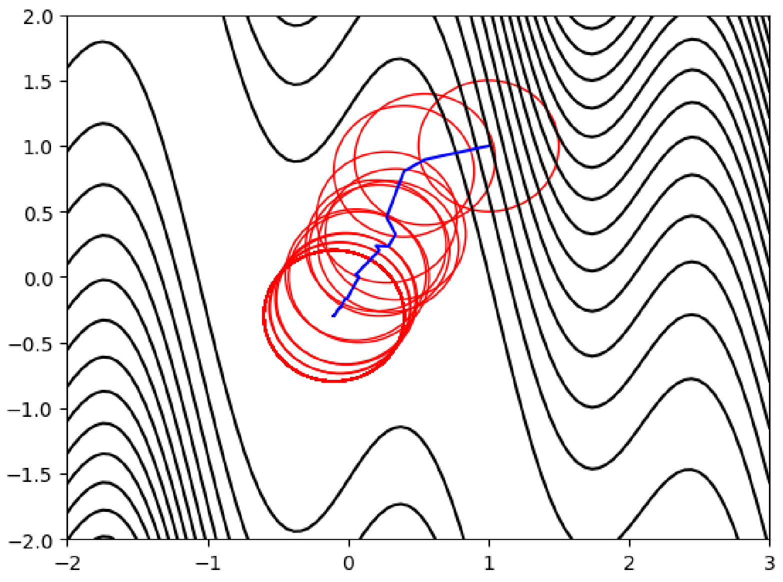

Fig. A1 Naive TR Method Radii Visualization - No noise in function (f) and no noise in gradient (g)

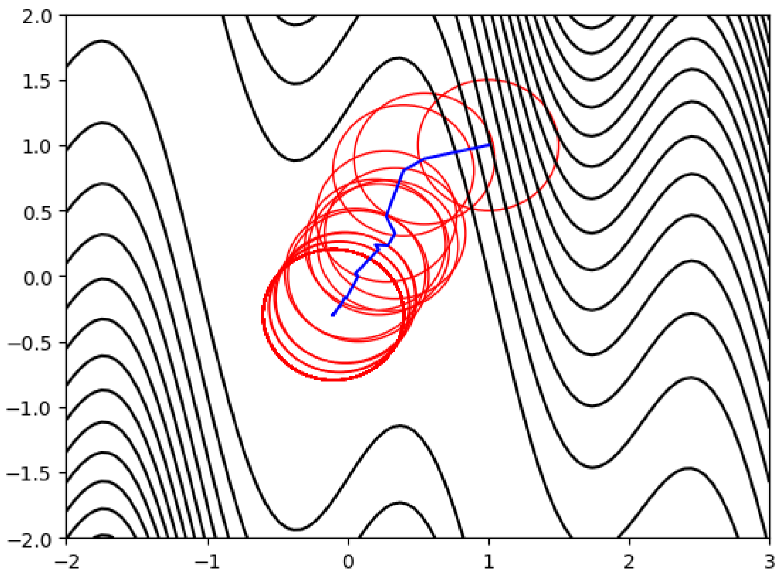

Fig. A2 Naive TR Method Radii Visualization - No noise in function (f) and no noise in gradient (g)

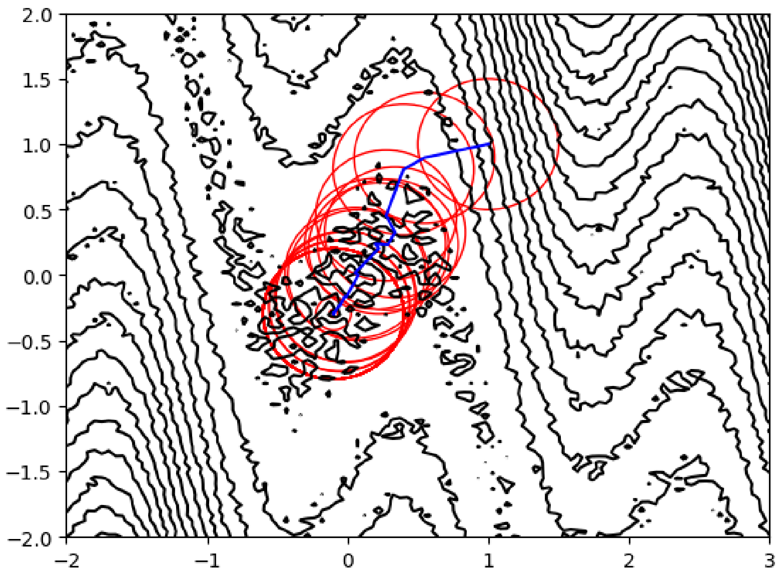

Fig. A3 Naive TR Method Radii Visualization - Noise in function ($\tilde{f}$) and no noise in gradient (g)

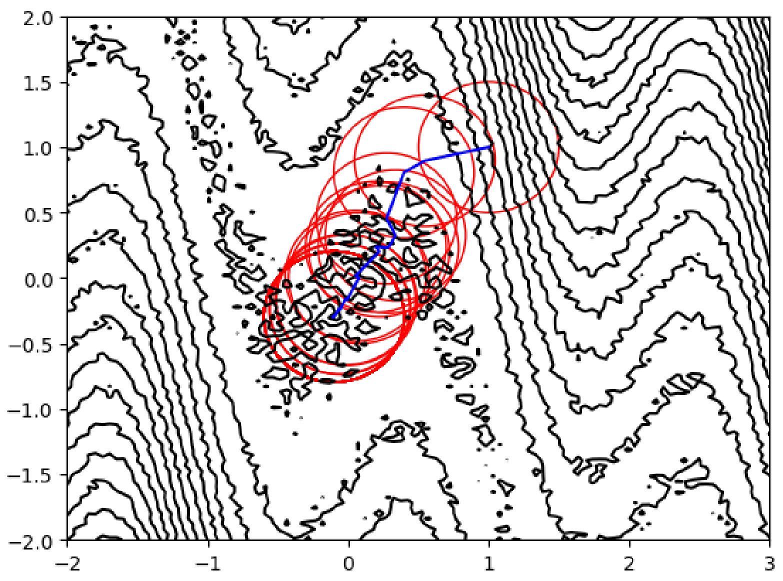

Fig. A4 Naive TR Method Radii Visualization - No noise in function ($\tilde{f}$) and no noise in gradient (g)

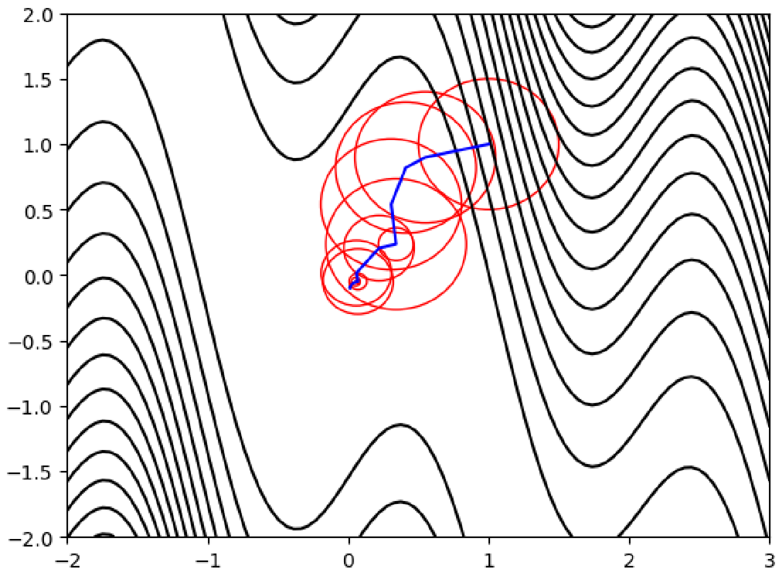

Fig. A5 Naive TR Method Radii Visualization - No noise in function (f) and noise in gradient ($\tilde{g}$)

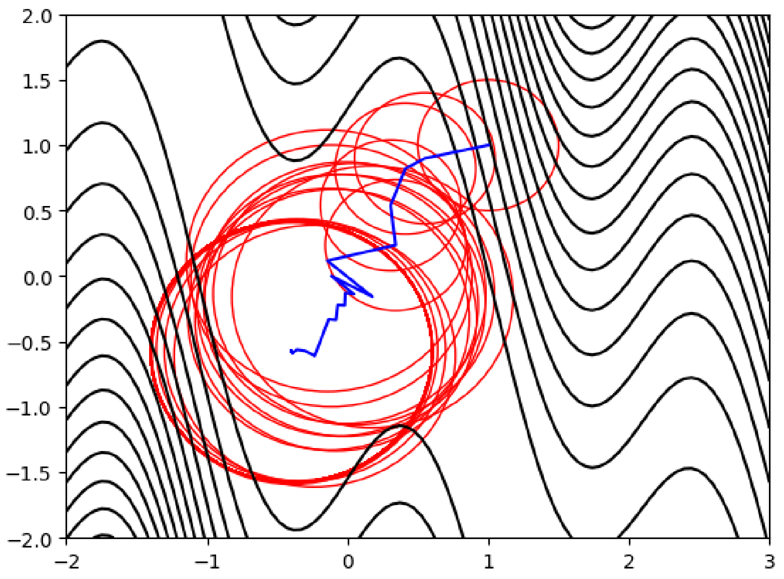

Fig. A6 Naive TR Method Radii Visualization - No noise in function (f) and noise in gradient ($\tilde{g}$)

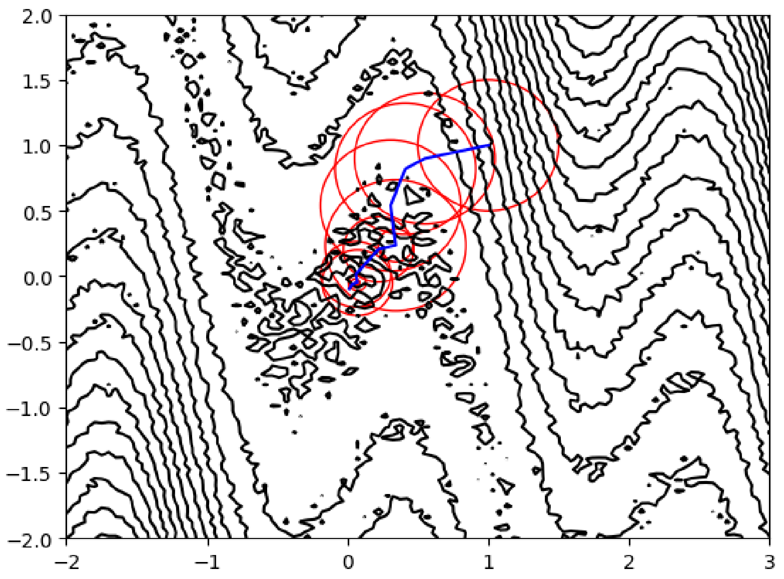

Fig. A7 Naive TR Method Radii Visualization - Noise in function ($\tilde{f}$) and noise in gradient ($\tilde{g}$)

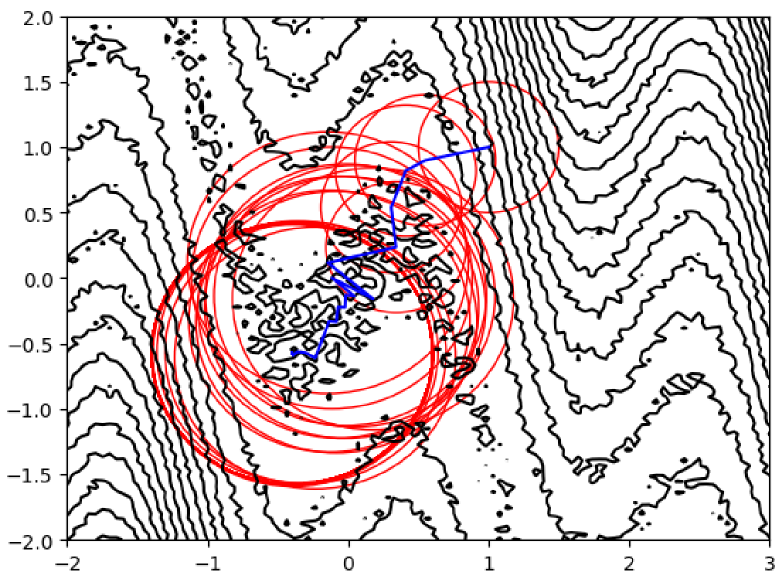

Fig. A8 Naive TR Method Radii Visualization - Noise in function ($\tilde{f}$) and noise in gradient ($\tilde{g}$)

References

[1] Sun, S., Nocedal, J.: A trust region method for the optimization of noisy functions (arXiv:2201.00973) https://doi.org/10.48550/arXiv.2201.00973 2201.00973 [math]. Accessed 2023-04-05

[2] Nocedal, J., Wright, S.J.: Numerical Optimization, 2e edn. Springer, New York, NY, USA (2006)

[3] Winfield, D.: Function minimization by interpolation in a data table **12**(3), 339–347 https://doi.org/10.1093/imamat/12.3.339 . Accessed 2023-04-08

[4] Xie, Y., Byrd, R., Nocedal, J.: Analysis of the BFGS method with errors (arXiv:1901.09063) https://doi.org/10.48550/arXiv.1901.09063 1901.09063 [math]. Accessed 2023-04-05