Greedy Newton: Newton's Method with Exact Line Search

Betty Shea[1] and Mark Schmidt[1,2]

[1]The University of British Columbia, Canada.
[2]Canada CIFAR AI Chair (Amii).

Contributing authors: sheaws@cs.ubc.ca; schmidtm@cs.ubc.ca;

Abstract

A defining characteristic of Newton's method is local superlinear convergence within a neighbourhood of a strict local minimum. However, outside this neighborhood Newton's method can converge slowly or even diverge. A common approach to dealing with non-convergence is using a step size that is set by an Armijo backtracking line search. With suitable initialization the line-search preserves local superlinear convergence, but may give sub-optimal progress when not near a solution. In this work we consider Newton's method under an exact line search, which we call "greedy Newton" (GN). We show that this leads to an improved global convergence rate, while retaining a local superlinear convergence rate. We empirically show that GN may work better than backtracking Newton by allowing significantly larger step sizes.

1 Introduction

For minimizing a twice-differentiable function $f : \mathbb{R}^n \to \mathbb{R}$, the pure Newton iteration x_{k+1}^N starting from some vector x_k is given by

$$x_{k+1}^N = x_k - \nabla^2 f(x_k)^{-1} \nabla f(x_k). \tag{1}$$

This method dates back to work by Newton and Raphson in the 1600s for finding roots of polynomials [1]. Since then, the method has evolved to work in a variety of settings. In optimization, Newton's method is a powerful tool for minimizing non-linear objectives mainly due to its remarkable property of superlinear (quadratic) convergence in a neighborhood of a strict local minimum under appropriate conditions [see 2, Section 9.5].

However, Newton's method also has known weaknesses. For example, the method is not guaranteed to converge in general or even decrease f. One of the standard fixes to the non-convergence is to introduce a step size α_k,

$$x_{k+1} = x_k - \alpha_k \nabla^2 f(x_k)^{-1} \nabla f(x_k). \tag{2}$$

The step size is typically set by first considering $\alpha_k = 1$ and dividing the step size by a fixed constant ("backtracking") until the Armijo condition is satisfied [2, Section 9.5]. Provided that we eventually get close enough to a strict local minimum, Armijo backtracking preserves the superlinear convergence of the pure Newton method. Many variations of Newton's method exist such as modifications for cases where $\nabla^2 f(x_k)$ is not invertible. Other variations include those based on trust-region methods instead of line searches [see 3], but the superlinear convergence proofs in the literature that we are aware of for (2) assume we first test $\alpha_k = 1$ and accept this step size if it satisfies a variant of the Armijo condition.

While $\alpha_k = 1$ becomes asymptotically optimal in the neighbourhood of a strict local minimizer, it may not be the optimal step size even when close to a minimizer. Further, when far away from a local minimizer using $\alpha_k = 1$ or the smaller values obtained by backtracking may converge slowly. This paper instead investigates Newton's method where the step size is set to minimize the function value,

$$\alpha_k \in \arg\min_{\alpha} f(x_k - \alpha \nabla^2 f(x_k)^{-1} \nabla f(x_k)). \tag{3}$$

We call using this exact line search within Newton's method the "greedy Newton" (GN) method. We first address two mis-conceptions the reader may have about this method:

- It typically **does not significantly increase the cost of Newton's method** to find a local minimizer of (3). For most problems just computing the Hessian is n-times more expensive than evaluating the function or directional derivatives. Thus, you can evaluate the objective in (3) and its derivative several times without changing the overall cost of the method. As an example, consider logistic regression with m training examples $\{a_i, b_i\}$ with dense features $a_i \in \mathbb{R}^n$ and binary labels $b_i \in \{-1, 1\}$,

  $$f(x) = \sum_{i=1}^{m} \log(1 + \exp(-b_i x^T a_i)). \tag{4}$$

  The cost of computing the Hessian for this problem is $O(mn^2)$, but the cost of evaluating f or a directional derivative of it is only $O(mn)$. With bisection we can solve the one-dimensional problem (3) to ϵ accuracy over a bounded domain in $O(\log(1/\epsilon))$ iterations, so the cost of a naive black-box numerical method is only $O(mn \log(1/\epsilon))$.

Further, if we exploit the linear composition structure of (4) the cost of bisection can be reduced to $O(mn + m\log(1/\epsilon))$. It is also possible to use faster one-dimensional minimizers like the secant method. Indeed, low-cost line searches are possible for a wide variety of problems including linear models, matrix factorization models, and certain neural networks [4–9].

- The **exact line search can yield a significantly smaller function value** than Armijo backtracking. The optimal step size (3) may be significantly larger than the maximum step size of 1 considered in standard implementations. While we could backtrack from a step size larger than 1, the Armijo condition itself can exclude the optimal step size. Indeed, for non-quadratic functions the *maximum step size allowed by the Armijo condition can be arbitrarily smaller* than the optimal step size.

By modifying standard arguments for the convergence of Newton's method, we show:

1. For strongly-convex functions with a Lipschitz-continuous gradient, GN slightly improves the global convergence rate compared to Armijo backtracking (Section 2.1).
2. Under the additional assumption that the Hessian is Lipschitz-continuous, superlinear convergence is achieved by any method (like GN) that decreases the function value by at least as much as the pure Newton iteration (1) (Section 2.2).
3. Under the same assumptions, we analyze the local convergence rate of Newton's method with arbitrary step sizes (Section 2.3).
4. The global convergence rate of Netwon's method can be further improved using hybrids with gradient descent (Section 2.4).

We are not aware of the superlinear convergence of GN appearing previously in the literature, although recent work bounds the GN step size for self-concordant functions [10] and various progress measures give superlinear convergence for solving non-linear equations [11]. We also note that asymptotic superlinear convergence of quasi-Newton methods with an exact line search is known classically [12] and concurrent with this work explicit superlinear rates of greedy quasi-Newton methods have been derived [13]. In Section 3 we experiment with the GN method for logistic regression. Our findings suggest that GN consistently works better than using Armijo backtracking, and substantially better for certain problems where the optimal step sizes can be much larger than 1.

2 Convergence of Greedy Newton Methods

All our results assume that f is twice-differentiable and that the eigenvalues of the Hessian $\nabla^2 f$ are bounded between positive constants μ and L for all x,

$$\mu I \preceq \nabla^2 f(x) \preceq LI. \tag{5}$$

These assumptions are equivalent to assuming that ∇f is L-Lipschitz continuous and that f is μ-strongly convex. We note that without these assumptions GN may not converge [14, 15], but various Hessian modifications guarantee convergence [see 3, Section 3.4]. The local superlinear convergence results also require that the Hessian is M-Lipschitz continuous,

$$\|\nabla^2 f(x) - \nabla^2 f(y)\| \leq M\|x - y\|, \tag{6}$$

where the matrix norm $\|\cdot\|$ on the left is the spectral norm while the vector norm on the right is the Euclidean norm. We give proofs of the results in this section in Appendix A.

2.1 Global Convergence of Greedy Newton

We first give a global rate of convergence for the GN method.

Proposition 1 *Let a twice-differentiable f be μ-strongly convex with an L-Lipschitz continuous gradient* (5)*. Then the iterations of Newton's method* (2) *with the greedy step size* (3) *satisfy*

$$f(x_k) - f(x_*) \leq \left(1 - \frac{\mu^2}{L^2}\right)^k [f(x_0) - f(x_*)].$$

This result implies that in order for the sub-optimality $d_k = (f(x_k) - f(x_*))$ to be less than ϵ, we require at most $(L^2/\mu^2)\log(d_0/\epsilon)$ iterations. If we instead set the step size by starting from a sufficiently large guess for α_k and halving it until the Armijo condition is satisfied, then with a sufficient decrease factor of $\sigma = 1/2$ we have a slower rate of

$$f(x_k) - f(x_*) \leq \left(1 - \frac{\mu^2}{2L^2}\right)^k [f(x_0) - f(x_*)].$$

This requires $2(L^2/\mu^2)\log(d_0/\epsilon)$ iterations to guarantee that we reach an accuracy of ϵ. Thus, GN halves the worst-case number of steps compared to this standard approach. If we use an Armijo sufficient decrease factor of $\sigma < 1/2$ and multiply the step size by $\beta < 1$ instead of $1/2$ when we backtrack, we require $(L^2/2\sigma\beta\mu^2)\log(d_0/\epsilon)$ [see 2, Section 9.5] (this again assumes the initial guess for α_k is sufficiently large, and note that it may need to be larger than 1). Note that the extra factor of $1/2\sigma\beta$ is greater than 1 since $\sigma\beta < 1/2$. Thus, GN performs as well as backtracking with an aribtrarily large initial α_k, and an arbitrarily small backtracking and sufficient decrease factor.

2.2 Local Convergence of "As Fast as Newton" Methods

We next consider a local rate for any method that decreases the function as much as the pure Newton method.

Proposition 2 *Let a twice-differentiable f be μ-strongly convex with an L-Lipschitz continuous gradient* (5)*, and an M-Lipschitz continuous Hessian* (6)*. Consider a method that* *is guaranteed to decrease the function as much as the pure Newton step* (1)*,* $f(x_{k+1}) \leq f(x_k - \nabla^2 f(x_k)^{-1}\nabla f(x_k))$*. The iterations of such methods satisfy*

$$\|x_{k+1} - x_*\| \leq \sqrt{\frac{L}{\mu}} \frac{M}{2\mu} \|x_k - x_*\|^2.$$

This result implies superlinear (quadratic) convergence beginning at the first iteration where we have

$$\|x_k - x_*\| < \sqrt{\frac{\mu}{L}} \frac{2\mu}{M}. \tag{7}$$

Note that this radius of fast convergence is smaller than the radius for the pure Newton method by a factor of $\sqrt{\mu/L}$ [16, 17], and thus we must be closer to the solution in order to guarantee superlinear convergence. Note that this result applies not only to the GN method but a variety of other possible methods.

2.3 Local Convergence of Newton with Arbitrary Step Size

We next consider a similar result, but for Newton's method with arbitrary step sizes.

Proposition 3 *Let a twice-differentiable* f *be* μ*-strongly convex with an* L*-Lipschitz continuous gradient* (5) *and an* M*-Lipschitz continuous Hessian* (6)*. Then Newton's method with a step size of* α_k (2) *satisfies*

$$\|x_{k+1} - x_*\| \leq |\alpha_k| \frac{M}{2\mu} \|x_k - x_*\|^2 + |\alpha_k - 1| \frac{L}{\mu} \|x_k - x_*\|$$

Note that if we assume $|\alpha_k - 1| \leq \|x_k - x_*\|$ then we have

$$\|x_{k+1} - x_*\| \leq |\alpha_k| \frac{M}{2\mu} \|x_k - x_*\|^2 + \frac{L}{\mu} \|x_k - x_*\|^2 = \frac{|\alpha_k| M + 2L}{2\mu} \|x_k - x_*\|^2.$$

Thus we have superliner (quadratic) convergence if for all k large enough we have

$$|\alpha_k - 1| \leq \|x_k - x_*\| \quad \text{and} \quad \|x_k - x_*\| < \frac{2\mu}{|\alpha_k| M + 2L}.$$

Thus, if L is similar to M and if α_k converges to 1 at least as fast as $\|x_k - x_*\|$ converges to zero, then Newton's method with non-zero step sizes has a similar radius of fast convergence to the pure Newton method. In the specific case of GN we have that α_k converges to 1 asymptotically as the quadratic approximation in the pure Newton method becomes exact. But the rate that α_k converges to 1 is less clear.

2.4 Global Convergence of Hybrid Gradient-Newton Methods

In Section 2.1 we review how GN improves on the linear convergence rate of Newton's method with backtracking from $(1 - 2\beta\sigma\mu^2/L^2)$ to $(1 - \mu^2/L^2)$. However, under the same assumptions gradient descent with an exact line search achieves a rate of $(1-\mu/L)$

while with backtracking gradient descent achieves a rate $(1 - 2\beta\sigma\mu/L)$ [see 2, Section 9.3]. Fortunately, it is possible to use the result of Section 2.2 to design methods that have these faster global linear convergence rates while maintaining a local superlinear convergence rate.

Perhaps the simplest hybrid method is the following:

- Let x_{k+1}^N be the pure Newton step (1) and x_{k+1}^G be the gradient descent step with exact line search,

$$x_{k+1}^G = x_k - \alpha_k^G \nabla f(x_k), \quad \alpha_k^G \in \arg\min_{\alpha} \{f(x_k - \alpha \nabla f(x_k))\}$$

- If $f(x_{k+1}^G) < f(x_{k+1}^N)$ take the gradient step, otherwise take the pure Newton step.

This approach guarantees the $(1 - \mu/L)$ linear rate is achieved at all iterations, while the result of Section 2.2 guarantees that this approach has a superlinear convergence rate. However, in our experiments this hybrid approach tended to perform worse than GN.

Other hybrid methods are possible, such as ones based on backtracking for either the gradient or Newton step. Another option discussed informally in the literature [18, 19] is to use a step size on both the gradient and Newton step,

$$x_{k+1} = x_k - \alpha_k^a \nabla f(x_k) - \alpha_k^b \nabla^2 f(x_k)^{-1} \nabla f(x_k),$$

and optimize the step sizes α_k^a and α_k^b. This "plane search" approach to setting two step sizes is efficient for many problems arising in machine learning [see 9]. However, in our experiments we found that this approach only gave small gains over the basic GN method (with α_k^a consistently being chosen close to zero).

3 Experiments

Our first experiment considers logistic regression (4) with the synthetic data included in the minFunc package [20]. This generates $m = 500$ examples where the elements of a_i are sampled from a standard normal, a true $\tilde{x}$ is sampled from a standard normal, and we set b_i to be the sign of $(a_i^T \tilde{x} + \delta_i)$ with δ_i sampled from a standard normal. We generated 4 versions: one with $n = 20$ yielding a strongly-convex problem, one with $n = 20$ where 10 of the features are repeated yielding a convex problem, one with $n = 200$ yielding a strictly convex problem, and one with $n = 2000$ yielding a convex problem. In the latter two cases the data is linearly separable, and thus no minimizing solution x^* exists. In these separable cases the function converges to its minimal value of 0 for vectors of the form κx as κ goes to ∞ for any vector x that separates the data. For the convex cases where the Hessian is not positive-definite we used $\nabla^2 f(x_k) + 10^{-12} I$ in place of the Hessian. We also considered L2-regularized variants of these problems with a regularization strength of $\lambda = 1$ (this makes all the problems strongly-convex with unique finite minimizers). We initialized all methods with the zero vector. For methods using the Armijo condition we used the standard choices of a sufficient decrease factor of $\sigma = 10^{-4}$ and a backtracking factor of $\beta = 1/2$.

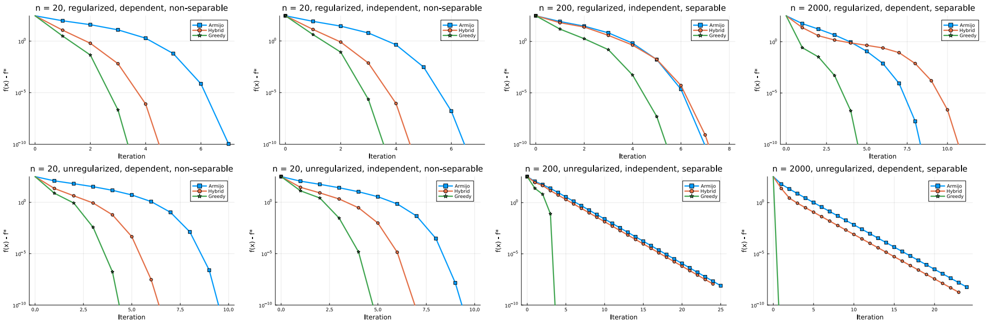


Fig. 1 Comparison of Newton with Armijo backtracking, hybrid gradient-Newton, and greedy Newton on logistic regression problems with regularization (top row) and without regularization (bottom row). From left to right, in the bottom row the problems are convex, strongly-convex, strictly-convex, and convex (all problems in the top row are strongly-convex). The right two datasets are linearly separable and the left two datasets are non-separable.

Although faster methods are possible, our experiments approximated an exact line search with the following simple bisection procedure for a descent direction d_k:

1. We initialize with $\alpha = 1$.
2. While the directional derivative $\nabla f(x_k + \alpha d_k)^T d_k$ is negative, we double α. Once we have an α large enough that the directional derivative is positive, this gives an interval $(0, \alpha)$ containing a minimizer.
3. Compute the mid-point of the interval, and cut the interval in half based on the directional derivative at the mid-point.
4. Repeat the previous step until the interval length is less than 10^{-8}.

We give the precise code used to implement GN for logistic regression in Appendix B.

In Figure 1 we compare GN with Newton's method with Armijo backtracking from an initial step size of 1, and the hybrid of greedy gradient descent and pure Newton method discussed in Section 2.4. We see that GN outperformed the other two methods in all settings. In Appendix C we report results on real data where the GN continued to perform the same or better than these other methods across 40 datasets. In Appendix D we repeat the synthetic data experiment but measure performance using runtime. We found that the GN method continued to outperform the other methods using the crude runtime measure of performance.

In Figure 2 we plot the step sizes used in the methods in Figure 1. For the hybrid method, a step size is 1 is reported when the pure Newton step is taken and otherwise the gradient descent step size is shown. Observe that GN used an initial step of at least 2 on every dataset, and the largest step size used was greater than 300.

The performance gain for GN was particularly large in the unregularized case for the two separable datasets (where the function is minimized by taking an x that separates the data and making its magnitude arbitrarily large); in these cases the Armijo and hybrid methods performed poorly while GN achieved numerical accuracy extremely quickly (in 4 iterations and 1 iteration respectively). For these datasets GN used step sizes much larger than 1.

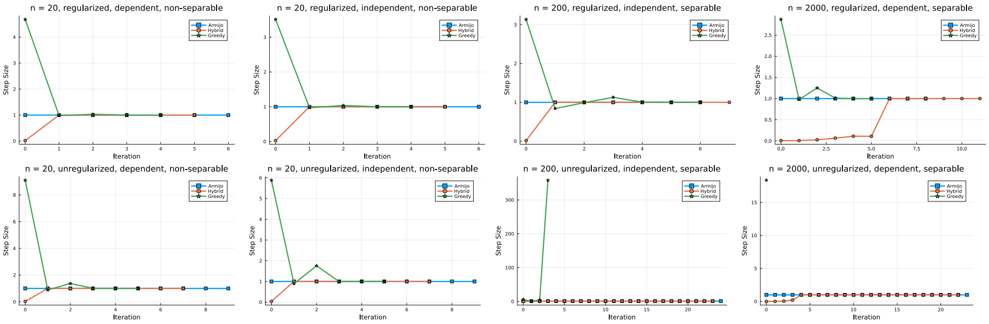


Fig. 2 Step size chosen on each iteration for the plots in Figure 1. For iterations where the hybrid method took a gradient step, the gradient descent step size is shown (when a pure Newton step was taken the step size is 1).

In our experiments on non-separable datasets or with regularization, GN initially takes a large step then immediately takes steps that become close to 1. In contrast, the Armijo backtracking method always accepted a step size of 1. Thus, GN's performance gain is largely due to using larger step sizes on early iterations. In light of this, one might consider modifying the Armijo line search to use a larger initialization. We explore this possibility in Appendix E. We found that in some cases initializing the Armijo backtracking procedure with a larger step could narrow or remove the initial gap between backtracking and GN. However, the Armijo backtracking procedure with a larger initialization often led to poor performance on later iterations (accepting large step sizes with worse performance than using a step size of 1).

4 Open Problems

Our experiments show that we can use Newton's method more advantageously when we do not restrict the step size to be less than 1. However, our theory does not reflect the large performance increases we saw in practice. Below we list some open problems:

1. Section 2.1: can we prove that step sizes bigger than 1 improve the global rate?
2. Section 2.2: is the additional $\sqrt{L/\mu}$ term in the superlinear rate necessary?
3. Section 2.3: can we analyze the rate at which α_k converges to 1?
4. Section 2.4: can we justify why GN outperforms the theoretically-faster hybrid method?
5. Section 3: can we prove a faster rate for GN on separable logistic regression problems?

We close by discussing using an exact line search in other Newton-like algorithms. First, we note that a precise step size search could also be added to Newton's method with cubic regularization and that this does not change the radius of superlinear convergence of that method (Appendix F). For practical quasi-Newton and Hessian-free Newton methods that avoid computing the Hessian explicitly [see 3], the cost of a precise step size search may significantly increase the iteration cost of the method. However, many problem structures arising in practice including logistic regression still allow an efficient line search even when function and gradient calculations are the bottleneck [see 9].

Appendix A Analysis of Greedy Newton

In this section, we prove the results in Sections 2.1-2.3. Our analyses are modifications of existing convergence analyses of the pure and backtracking Newton method [2, 17, 21] to use an exact line search.

A.1 Global Convergence of Greedy Newton

In this section we give the proof of Proposition 2.1. Our assumption is that $\nabla^2 f(x_k)$ is positive definite with eigenvalues in $[\mu, L]$. This implies that $\nabla^2 f(x_k)^{-1}$ is symmetric and positive definite with eigenvalues in $[1/L, 1/\mu]$. Using these facts in a Taylor expansion gives

$$\begin{aligned}
f(x_{k+1}) &= f(x_k) + \nabla f(x_k)^T(x_{k+1}-x_k) + \frac{1}{2}(x_{k+1}-x_k)^T\nabla^2 f(z)(x_{k+1}-x_k) && \text{(for z between x_{k+1} and x_k)}\\
&\leq f(x_k) + \nabla f(x_k)^T(x_{k+1}-x_k) + \frac{L}{2}\|x_{k+1}-x_k\|^2 && (\nabla^2 f(z) \preceq LI)\\
&= f(x_k) - \alpha_k \nabla f(x_k)^T\nabla^2 f(x_k)^{-1}\nabla f(x_k) + \frac{L\alpha_k^2}{2}\|\nabla^2 f(x_k)^{-1}\nabla f(x_k)\|^2 && \text{(x_{k+1} is Newton step (2))}\\
&= f(x_k) - \alpha_k \nabla f(x_k)^T\nabla^2 f(x_k)^{-1}\nabla f(x_k) + \frac{L\alpha_k^2}{2}\nabla f(x_k)^T\nabla^2 f(x_k)^{-2}\nabla f(x_k)\\
&\leq f(x_k) - \alpha_k \nabla f(x_k)^T\nabla^2 f(x_k)^{-1}\nabla f(x_k) + \frac{L\alpha_k^2}{2\mu}\nabla f(x_k)^T\nabla^2 f(x_k)^{-1}\nabla f(x_k) && (\nabla^2 f(x_k)^{-1} \preceq (1/\mu)I)\\
&= f(x_k) - \alpha_k\left(1 - \frac{\alpha_k}{2}\frac{L}{\mu}\right)\nabla f(x_k)^T\nabla^2 f(x_k)^{-1}\nabla f(x_k)
\end{aligned}$$

With an exact line search (3), we decrease the function by at least as much as choosing $\alpha_k = \mu/L$, so we have

$$\begin{aligned}
f(x_{k+1}) &\leq f(x_k) - \frac{\mu}{2L}\nabla f(x_k)^T\nabla^2 f(x_k)^{-1}\nabla f(x_k)\\
&\leq f(x_k) - \frac{\mu}{2L^2}\|\nabla f(x_k)\|^2,
\end{aligned}$$

using that $\nabla^2 f(x_k)^{-1} \succeq (1/L)I$. By strong convexity we have $\frac{1}{2\mu}\|\nabla f(x_k)\|^2 \geq f(x_k) - f(x_*)$, using this and subtracting $f(x_*)$ from both sides gives

$$\begin{aligned}
f(x_{k+1}) - f^* &\leq f(x_k) - f(x_*) - \frac{\mu^2}{L^2}[f(x_k) - f(x_*)]\\
&= \left(1 - \frac{\mu^2}{L^2}\right)[f(x_k) - f^*].
\end{aligned}$$

Applying this recursively gives the result.

A.2 Local Convergence of "As Fast as Newton" Methods

In this section we give the proof of Proposition 2.2. If f is μ-strongly convex with an M-Lipschitz Hessian, then the pure Newton update x_{k+1}^N (1) satisfies [17, 22]

$$\|x_{k+1}^N - x_*\| \leq \frac{M}{2\mu}\|x_k - x_*\|^2 \tag{A1}$$

If ∇f is L-Lipschitz then we have

$$f(x) \leq f(x_*) + \nabla f(x_*)^\intercal (x - x_*) + \frac{L}{2}\|x - x_*\|^2,$$

which using $\nabla f(x_*) = 0$ implies

$$f(x) - f(x_*) \leq \frac{L}{2}\|x - x_*\|^2 \tag{A2}$$

Similarly, μ-strongly convexity of f implies that

$$f(x) - f(x_*) \geq \frac{\mu}{2}\|x - x_*\|^2 \tag{A3}$$

Thus, if an algorithm has $f(x_{k+1}) \leq f(x_{k+1}^N)$ then we have

$$\begin{aligned}\|x_{k+1} - x_*\|^2 &\leq \frac{2}{\mu}[f(x_{k+1}) - f(x_*)] \\ &\leq \frac{2}{\mu}[f(x_{k+1}^N) - f(x_*)] \\ &\leq \frac{L}{\mu}\|x_{k+1}^N - x_*\|^2.\end{aligned}$$

Combined with the progress of Newton's method given by Equation (A1), we get

$$\|x_{k+1} - x_*\| \leq \sqrt{\frac{L}{\mu}}\frac{M}{2\mu}\|x_k - x_*\|^2 \tag{A4}$$

A.3 Local Convergence of Newton with Arbitrary Step Size

In this section we give the proof of Proposition 2.3. The result holds if $\alpha_k = 0$ since $L/\mu \geq 1$. Thus, we focus on the case of $\alpha_k \neq 0$. Subtracting x_* from both sides of the Newton update with a step size of $\alpha_k \neq 0$ (2) gives

$$x_{k+1} - x_* = x_k - \alpha_k \nabla^2 f(x_k)^{-1}\nabla f(x_k) - x_*$$

Since $\nabla f(x_*) = 0$, we can add the quantity $\alpha_k \nabla^2 f(x_k)^{-1} \nabla f(x_*)$ on the right hand side and rearrange

$$\begin{aligned} x_{k+1} - x_* &= (x_k - x_*) - \alpha_k \nabla^2 f(x_k)^{-1} \left(\nabla f(x_k) - \nabla f(x_*)\right) \\ &= \frac{\alpha_k}{\alpha_k} \nabla^2 f(x_k)^{-1} \nabla^2 f(x_k)(x_k - x_*) - \alpha_k \nabla^2 f(x_k)^{-1} \left(\nabla f(x_k) - \nabla f(x_*)\right) \end{aligned}$$

to get

$$x_{k+1} - x_* = \alpha_k \nabla^2 f(x_k)^{-1} \left[\frac{1}{\alpha_k} \nabla^2 f(x_k)(x_k - x_*) - (\nabla f(x_k) - \nabla f(x_*)) \right] \tag{A5}$$

From Taylor's theorem we have

$$\nabla f(x_*) = \nabla f(x_k + (x_* - x_k)) = \nabla f(x_k) + \int_0^1 \nabla^2 f(x_k + t(x_* - x_k))(x_* - x_k) dt$$

$$\implies \nabla f(x_k) - \nabla f(x_*) = \int_0^1 \nabla^2 f(x_k + t(x_* - x_k))(x_k - x_*) dt$$

Substituting this into (A5) gives

$$\begin{aligned} x_{k+1} - x_* &= \alpha_k \nabla^2 f(x_k)^{-1} \left[\frac{1}{\alpha_k} \nabla^2 f(x_k)(x_k - x_*) - \int_0^1 \nabla^2 f(x_k + t(x_* - x_k))(x_k - x_*) dt \right] \\ &= \alpha_k \nabla^2 f(x_k)^{-1} \left[\frac{1}{\alpha_k} \nabla^2 f(x_k) - \int_0^1 \nabla^2 f(x_k + t(x_* - x_k)) dt \right] (x_k - x_*) \\ &= \alpha_k \nabla^2 f(x_k)^{-1} \left[\int_0^1 \left(\frac{1}{\alpha_k} \nabla^2 f(x_k) - \nabla^2 f(x_k + t(x_* - x_k)) \right) dt \right] (x_k - x_*). \end{aligned}$$

Taking norms on both sides and using the Cauchy-Schwartz inequality gives

$$\|x_{k+1} - x_*\| \leq |\alpha_k| \|\nabla^2 f(x_k)^{-1}\| \left\| \int_0^1 \frac{1}{\alpha_k} \nabla^2 f(x_k) - \nabla^2 f(x_k + t(x_* - x_k)) dt \right\| \|x_k - x_*\| \tag{A6}$$

We bound the factor containing the integral using the triangle inequality, $\| \int_a^b g(t) dt \| \leq \int_a^b \|g(t)\| dt$,

$$\left\| \int_0^1 \frac{1}{\alpha_k} \nabla^2 f(x_k) - \nabla^2 f(x_k + t(x_* - x_k)) dt \right\| \leq \int_0^1 \left\| \frac{1}{\alpha_k} \nabla^2 f(x_k) - \nabla^2 f(x_k + t(x_* - x_k)) \right\| dt$$

Using Lipschitz continuity of the gradient (5) and Hessian (6), this could be rewritten

$$\int_0^1 \left\| -\nabla^2 f(x_k) + \frac{1}{\alpha_k} \nabla^2 f(x_k) + \nabla^2 f(x_k) - \nabla^2 f(x_k + t(x_* - x_k)) \right\| dt$$

$$
\begin{aligned}
&= \int_0^1 \left\| \left(-1 + \frac{1}{\alpha_k}\right) \nabla^2 f(x_k) + \big(\nabla^2 f(x_k) - \nabla^2 f(x_k + t(x_* - x_k)\big) \right\| dt \\
&\le \int_0^1 \left|1 - \frac{1}{\alpha_k}\right| \left\|\nabla^2 f(x_k)\right\| + \left\|\nabla^2 f(x_k) - \nabla^2 f(x_k + t(x_* - x_k)\right\| dt \\
&\le \int_0^1 \left|1 - \frac{1}{\alpha_k}\right| \left\|\nabla^2 f(x_k)\right\| + Mt\|x_k - x_*\| dt \\
&\le \int_0^1 \left|1 - \frac{1}{\alpha_k}\right| L + Mt\|x_k - x_*\| dt,
\end{aligned}
$$

Substituting this back into (A6) gives

$$
\begin{aligned}
\|x_{k+1} - x_*\| &\le |\alpha_k| \|\nabla^2 f(x_k)^{-1}\| \int_0^1 \left[\left|1 - \frac{1}{\alpha_k}\right| L + Mt\|x_k - x_*\|\right] dt \|x_k - x_*\| \\
&= |\alpha_k| \|\nabla^2 f(x_k)^{-1}\| \left[\left|1 - \frac{1}{\alpha_k}\right| L + \frac{M}{2}\|x_k - x_*\|\right] \|x_k - x_*\| \\
&\le \frac{|\alpha_k|}{\mu} \left[\left|1 - \frac{1}{\alpha_k}\right| L + \frac{M}{2}\|x_k - x_*\|\right] \|x_k - x_*\| && \text{(by strong convexity)} \\
&= |\alpha_k - 1| \frac{L}{\mu} \|x_k - x_*\| + |\alpha_k| \frac{M}{2\mu} \|x_k - x_*\|^2.
\end{aligned}
$$

Appendix B Greedy Newton Implementation

In Figure B1 we give our Julia code implementation of the GN method, as well as the functions required to apply the code to logistic regression.

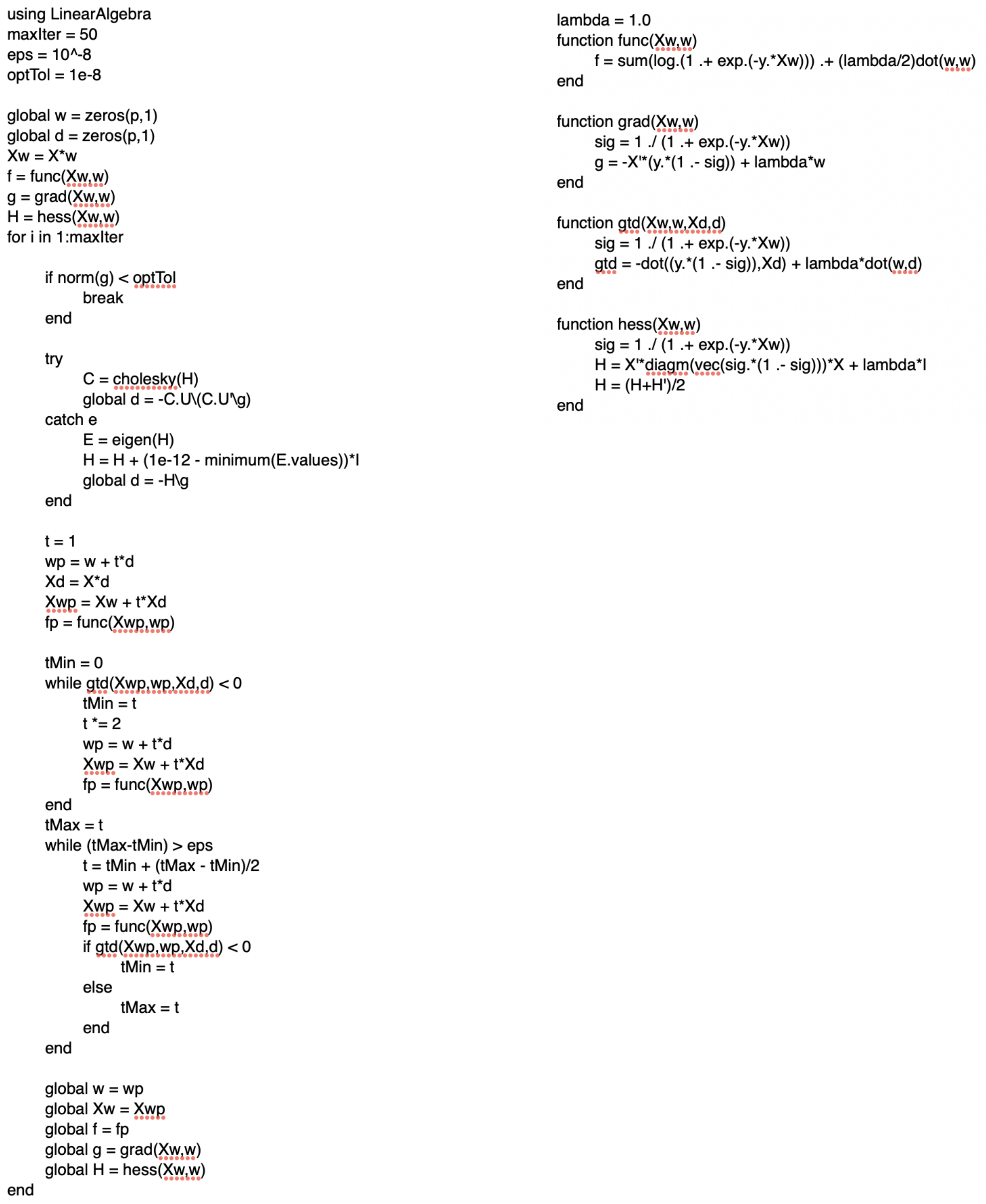
Appendix C Real Data Experiments

We performed logistic regression experiments on over 40 datasets obtained using the Dataset Downloader software (https://github.com/fKunstner/dataset-downloader). On the majority of these datasets, we observed the following trends whether we regularized or not:

- The Armijo and hybrid methods produced identical iterations (no gradient descent steps were selected).
- The GN method outperformed the other methods.
- The GN method typically used a large initial step size but the step sizes quickly converged to 1.

In Figures C2 and C3 we plot the performance on 8 datasets where we observed this typical case. On a smaller number of datasets, we observed different behaviours including:

- Cases where the performance of the hybrid method was better or worse than the Armijo method.
- Cases where all methods converged extremely quickly.



```
using LinearAlgebra
maxIter = 50
eps = 10^-8
optTol = 1e-8

global w = zeros(p,1)
global d = zeros(p,1)
Xw = X*w
f = func(Xw,w)
g = grad(Xw,w)
H = hess(Xw,w)
for i in 1:maxIter

    if norm(g) < optTol
        break
    end

    try
        C = cholesky(H)
        global d = -C.U\(C.U'\g)
    catch e
        E = eigen(H)
        H = H + (1e-12 - minimum(E.values))*I
        global d = -H\g
    end

    t = 1
    wp = w + t*d
    Xd = X*d
    Xwp = Xw + t*Xd
    fp = func(Xwp,wp)

    tMin = 0
    while gtd(Xwp,wp,Xd,d) < 0
        tMin = t
        t *= 2
        wp = w + t*d
        Xwp = Xw + t*Xd
        fp = func(Xwp,wp)
    end
    tMax = t
    while (tMax-tMin) > eps
        t = tMin + (tMax - tMin)/2
        wp = w + t*d
        Xwp = Xw + t*Xd
        fp = func(Xwp,wp)
        if gtd(Xwp,wp,Xd,d) < 0
            tMin = t
        else
            tMax = t
        end
    end

    global w = wp
    global Xw = Xwp
    global f = fp
    global g = grad(Xw,w)
    global H = hess(Xw,w)
end
```

```
lambda = 1.0
function func(Xw,w)
    f = sum(log.(1 .+ exp.(-y.*Xw))) .+ (lambda/2)dot(w,w)
end

function grad(Xw,w)
    sig = 1 ./ (1 .+ exp.(-y.*Xw))
    g = -X'*(y.*(1 .- sig)) + lambda*w
end

function gtd(Xw,w,Xd,d)
    sig = 1 ./ (1 .+ exp.(-y.*Xw))
    gtd = -dot((y.*(1 .- sig)),Xd) + lambda*dot(w,d)
end

function hess(Xw,w)
    sig = 1 ./ (1 .+ exp.(-y.*Xw))
    H = X'*diagm(vec(sig.*(1 .- sig)))*X + lambda*I
    H = (H+H')/2
end
```

Fig. B1 On the left is the Julia code used in our experiments implemented the greedy Newton method. On the right is our code for regularized logistic regression implementing the function value, gradient, directional derivative, and Hessian. The unregularized version corresponds to setting lambda to 0 (or equivalently removing the terms involving lambda). Note that the bottlenecks on each iteration of the GN code are the matrix operations involving the Hessian in the try-catch statement, computing the matrix-vector product Xd, and finally calling the gradient and Hessian functions at the end. All other operations are performed on vectors or scalars.

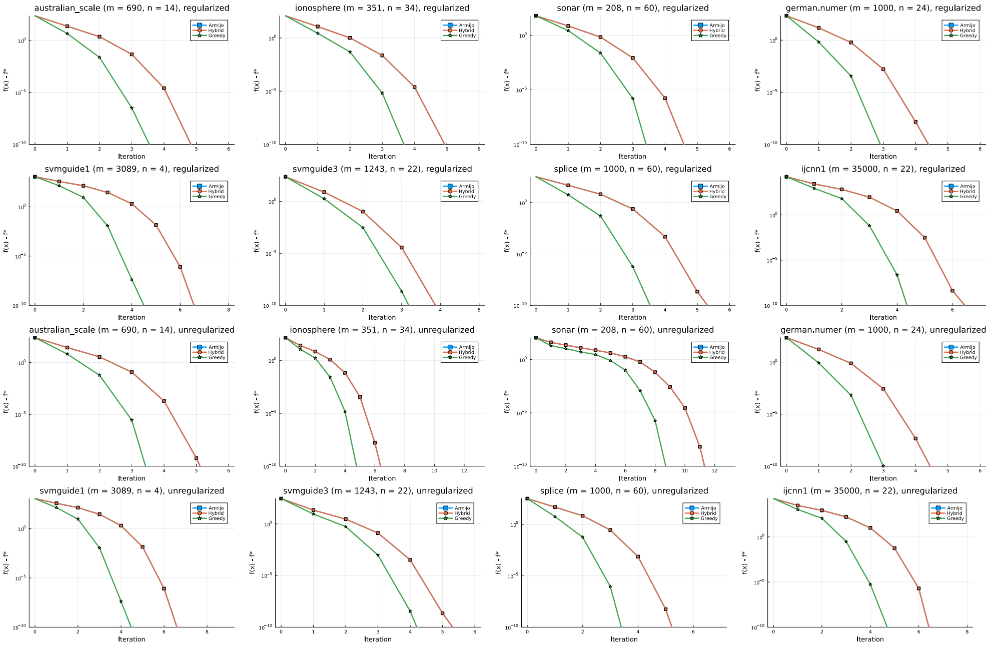


Fig. C2 Comparison of methods on real logistic regression datasets, in 8 cases where we observed typical performance.

- Cases where the hybrid method performed similar to greedy.
- Cases where the GN method converges in 1 step (and one case where the hybrid method did this).
- Cases where the GN method eventually begins to oscillate between two non-unit step sizes (this seemed to happen for problems with singular Hessians, so is likely due to the particular Hessian modification strategy we used).

In Figures C4 and C5 we plot the performance on 8 datasets where we observed some of these atypical behaviours. Despite these different behaviours, we note that GN performed the same or better than the other methods across all datasets.

Appendix D Runtime Experiments

In Figure D6 we repeat the synthetic data experiments of Figure 1 but measure performance in terms of runtime. The GN method still outperformed the other methods in terms of runtime. In some cases the hybrid method performed worse in terms of runtime, although it is not clear why (other than requiring an additional matrix-vector product per iteration compared to the other methods). We caution the reader that runtime comparisons are very sensitive to precise implementations, and that for larger datasets the iteration counts of Figure 1 are likely to be a better predictor of performance than runtime performance on specific datasets with specific hardware.

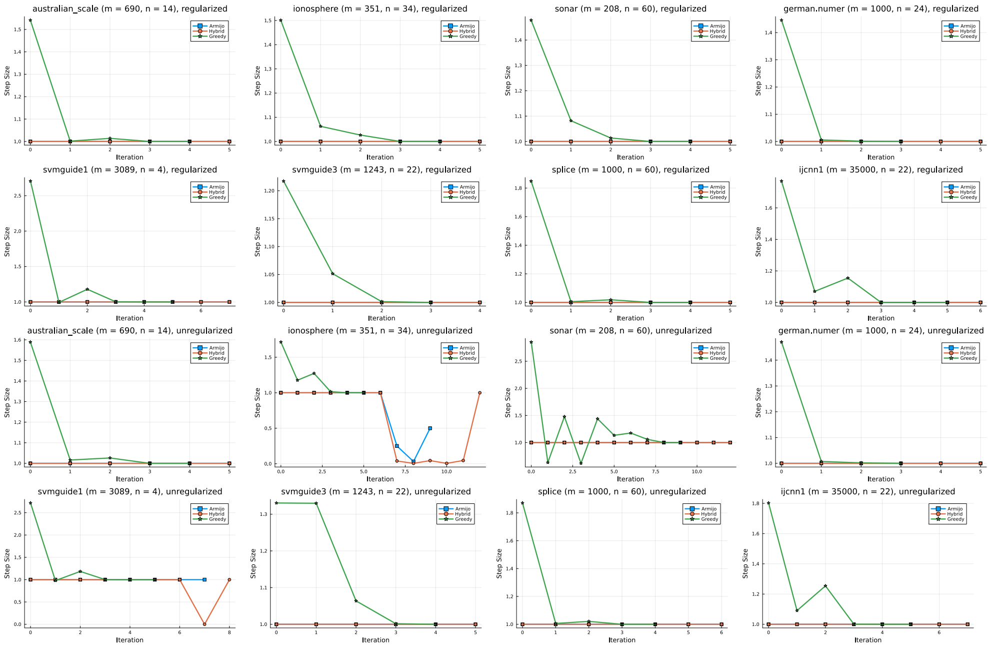


Fig. C3 Step sizes of methods on real logistic regression datasets, in 8 cases where we observed typical performance.

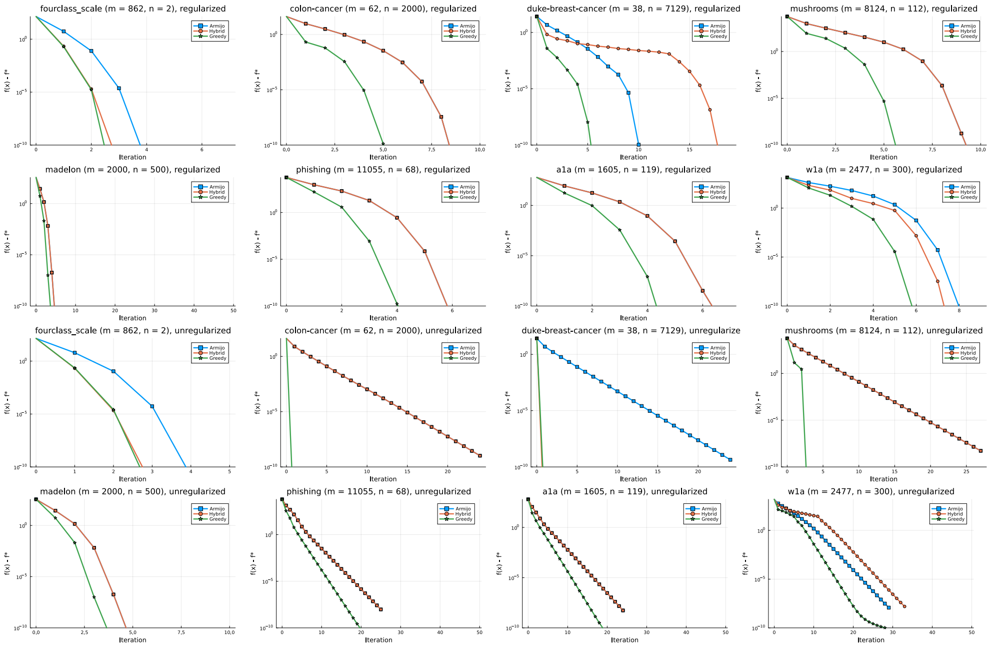


Fig. C4 Comparison of methods on real logistic regression datasets, in 8 cases where we observed atypical performance in either the regularized or unregularized setting.

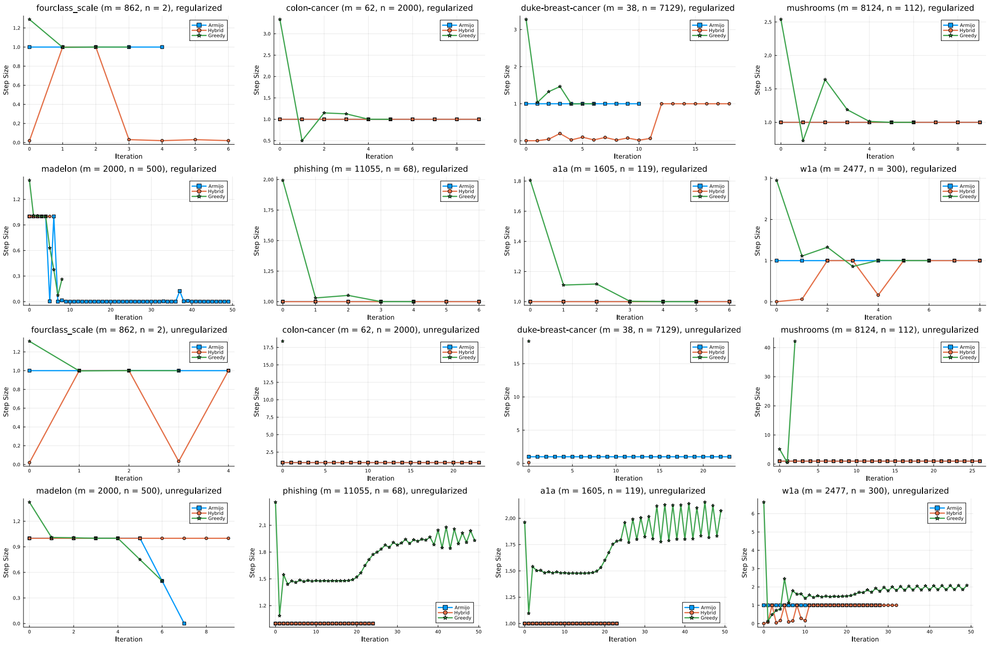


Fig. C5 Step sizes of methods on real logistic regression datasets, in 8 cases where we observed atypical performance in either the regularized or unregularized setting.

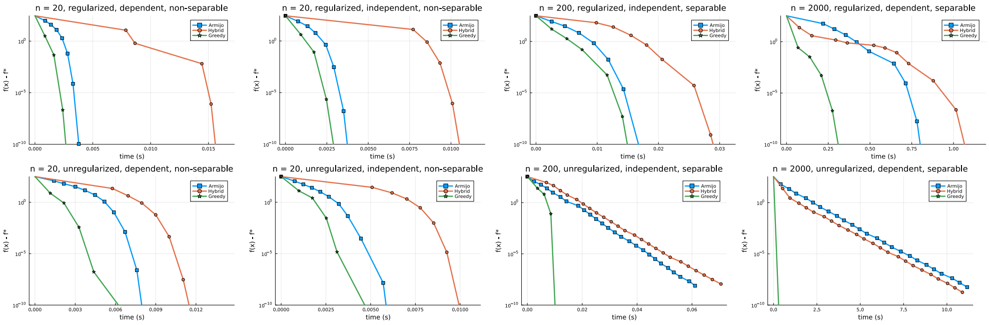


Fig. D6 Runtime comparison of Newton with Armijo backtracking, hybrid gradient-Newton, and greedy Newton on logistic regression problems with regularization (top row) and without regularization (bottom row).

Appendix E Armijo Backtracking with Larger Initialization

If Figure E7 we repeat our synthetic experiments but explore different initializations of the Armijo backtracking line-search. In addition to the standard initialization of $\alpha = 1$, we explored using a slightly larger initialization of $\alpha = 2$ and a much larger initialization of $\alpha = 8$. Note that $\alpha = 8$ is larger than all the step sizes chosen by GN across iterations, except on the two unregularized seperable datasets. While these

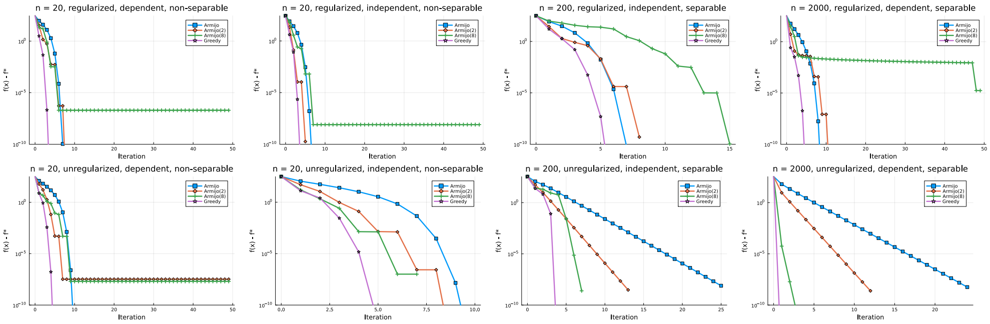


Fig. E7 Comparison of greedy Newton to Newton with Armijo backtracking using three choices for the step size initialization (1, 2, and 8). We see that larger initializations can narrow or remove the gap with greedy Newton on early iterations, but can lead to poor performance on later iterations.

larger step sizes led to better performance on early iterations for many datasets, for some datasets they eventually led to much worse performance (due to using step sizes larger than 1 on later iterations).

Appendix F Greedy Newton with Cubic Regularization

Given the Lipschitz constant of the Hessian M, Newton's method with cubic regularization [16] uses iterations of the form

$$x_{k+1}^{C,M} \in \arg\min_y \left\{ f(x_k) + \nabla f(x_k)^T(y - x_k) + \frac{1}{2}(y - x_k)^T \nabla^2 f(x_k)(y - x_k) + \frac{M}{6}\|y - x_k\|^3 \right\}.$$

Unlike the classic Newton method, it is known that this method has a quadratic convergence rate directly in terms of function values. In particular, the method converges quadratically beginning from the first iteration where $f(x_k) - f(x_*) \leq \frac{M}{2\mu}$ [see 21, Section 4.2.6].

There are several ways we could add a step size to this method:

1. If M is known, we could use iterations that take a step in the direction of a solution of the cubic sub-problem,
$$x_{k+1} = x_k + \alpha_k(x_{k+1}^{C,M} - x_k), \tag{F7}$$
where α_k is chosen to minimize the function value. The cubic step corresponds to choosing $\alpha_k = 1$, but other values may decrease the objective function by a larger amount. Because we have $f(x_{k+1}) \leq f(x_k^C)$ with this method, it has the same radius of superlinear convergence as the basic Newton's method with cubic regularization.
2. If M is not known, it is common to use a backtracking procedure to set M [16]. In this case, we could alternate between backtracking to find an M guaranteeing sufficient progress, and doing a line search on α_k in the direction of a solution of the cubic sub-problem (F7) with the current approximation of M.

3. If M is not known, we could alternately search for the M that minimizes the function value,

$$M_k \in \arg\min_{M} \left\{ f(x_{k+1}^{C,M}) \right\},$$

and then use this M_k in place of M in the cubic update. A re-formulation of this update is using [16]

$$x_{k+1} = x_k - (\nabla^2 f(x_k)^{-1} + \lambda I)^{-1} \nabla f(x_k),$$

and choosing

$$\lambda_k \in \arg\min_{\lambda} f(x_k - (\nabla^2 f(x_k)^{-1} + \lambda I)^{-1} \nabla f(x_k)),$$

which is a greedy version of a variation on the classic Levenberg-Marquardt update. This method decreases the function at least as much as any particular choice of M. Thus, it preserves the region of superlinear convergence without requiring us to know M and would likely perform better in practice than using a fixed M. The disadvantage of this approach compared to line search methods is that it involves additional operations with the Hessian. However, given the gradient and Hessian we could use a suitable factorization of the Hessian such as the Schur factorization or singular value decomposition [23] to implement this search in $O(n^3 + n^2 \log(1/\epsilon))$ time plus the cost of evaluating f $O(\log(1/\epsilon))$ times.

Acknowledgements. We thank Frederik Kunstner and Nicolas Boumal for valuable discussions, and to the anonymous reviewers for suggestions that improved the paper.

Funding. Betty Shea is funded by an NSERC Canada Graduate Scholarship. The work was partially supported by the Canada CIFAR AI Chair Program and NSERC Discovery Grant RGPIN-2022-036669.

Data availability. The datasets used in this paper are publicly available at the minFunc package [20] available at https://www.cs.ubc.ca/~schmidtm/Software/minFunc.html.

Conflict of interest. The authors have no financial or proprietary interests in any material discussed in this article.

References

[1] Deuflhard, P.: A short history of Newton's method. Documenta Mathematica, Optimization stories, 25–30 (2012)

[2] Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge, ??? (2004)

[3] Nocedal, J., Wright, S.J.: Numerical Optimization, 2nd Ed. Springer, ??? (2006)

[4] Narkiss, G., Zibulevsky, M.: Sequential subspace optimization method for large-scale unconstrained problems. Technical report, Technion - Israel Institute of Technology (2005)

[5] Narkiss, G., Zibulevsky, M.: Support vector machine via sequential subspace optimization. Technical report, Technion - Israel Institute of Technology (2005)

[6] Zibulevsky, M.: SESOP-TN: combining sequential subspace optimization with truncated Newton method (2008)

[7] Zibulevsky, M.: SESOP_PACK: Matlab tool for sequential subspace optimization methods (2010)

[8] Sorber, L., Domanov, I., Van Barel, M., De Lathauwer, L.: Exact line and plane search for tensor optimization. Computational Optimization and Applications **63**, 121–142 (2016)

[9] Shea, B., Schmidt, M.: Why line search when you can plane search? SO-friendly neural networks allow per-iteration optimization of learning and momentum rates for every layer. arXiv preprint arXiv:2406.17954 (2024)

[10] Ivanova, A., Hildebrand, R.: Optimal step length for the maximal decrease of a self-concordant function by the Newton method. Optimization Letters, 1–8 (2023)

[11] Burdakov, O.P.: Some globally convergent modifications of Newton's method for solving systems of nonlinear equations. Doklady Akademii Nauk **254**(3), 521–523 (1980). Russian Academy of Sciences

[12] Powell, M.J.: On the convergence of the variable metric algorithm. IMA Journal of Applied Mathematics **7**(1), 21–36 (1971)

[13] Jin, Q., Jiang, R., Mokhtari, A.: Non-asymptotic global convergence rates of bfgs with exact line search. arXiv preprint arXiv:2404.01267 (2024)

[14] Mascarenhas, W.F.: On the divergence of line search methods. Computational & Applied Mathematics **26**, 129–169 (2007)

[15] Jarre, F., Toint, P.L.: Simple examples for the failure of Newton's method with

line search for strictly convex minimization. Mathematical Programming **158**(1-2), 23–34 (2016)

[16] Nesterov, Y., Polyak, B.: Cubic regularization of Newton method and its global performance. Math. Program. **108**, 177–205 (2006)

[17] Sun, Y.: The Happy Optimist: Newton's Method I (2021)

[18] Conn, A.R., Toint, P.L., Sartenaer, A., Gould, N.: On iterated-subspace minimization methods for nonlinear optimization. Technical report, P00024646 (1994)

[19] Zibulevsky, M.: Speeding-up convergence via sequential subspace optimization: Current state and future directions. arXiv preprint arXiv:1401.0159 (2013)

[20] Schmidt, M.: Minfunc: unconstrained differentiable multivariate optimization in Matlab (2005)

[21] Nesterov, Y.: Lectures on Convex Optimization, 2nd Ed. Springer, ??? (2018)

[22] Bertsekas, D.P.: Nonlinear Programming, 3rd edn. Athena Scientific, ??? (2016)

[23] Golub, G.H., Van Loan, C.F.: Matrix Computations. JHU press, ??? (2013)