1 STOCHASTIC-GRADIENT LD

$$h = \frac{N}{n}\sum_{i=1}^{n}\nabla \log p(\mathbf{x}_i|\boldsymbol{\theta}) - \sum_{i=1}^{N}\nabla \log p(\mathbf{x}_i|\boldsymbol{\theta}) \quad (1)$$

$$= \frac{N}{n}\sum_{i=1}^{n} s_i \quad (2)$$

$$s_i = \nabla \log p(\mathbf{x}_i|\boldsymbol{\theta}) + \frac{1}{N}\nabla p(\boldsymbol{\theta}) \quad (3)$$

$$\mathrm{V}(h) = \frac{N^2}{n^2}\sum_{i=1}^{n}\mathrm{Var}(s_i) \quad (4)$$

$$= \frac{N^2}{n^2}\sum_{i=1}^{n}\frac{1}{n}\left(\sum_{j=1}(s_i - \bar{s})^2\right) \quad (5)$$

$$= \frac{N^2}{n^2}\sum_{i=1}^{n} V_s \quad (6)$$

$$= \frac{N^2}{n^2} n V_s \quad (7)$$

$$= \frac{N^2}{n} V_s \quad (8)$$

$$\mathrm{Var}(\boldsymbol{\theta}) = \frac{\epsilon^2}{4}\mathrm{Var}(h) + \epsilon \quad (9)$$

$$\frac{\epsilon^2}{4}\mathrm{Var}(h) << \epsilon \quad (10)$$

$$\frac{\epsilon N^2}{4n} V_s << 1 \quad (11)$$

2 SGLD-ABC

2.1 ABC Cost Function

For optimization purposes we can search for the maximum a posteriori (MAP) estimator $\hat{\boldsymbol{\theta}}$ using the log-joint distribution $\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \mathcal{MAP}$, where $\mathcal{MAP}$ is

$$\mathcal{LJ} = \log \pi(\boldsymbol{\theta}) + \log \pi(\mathbf{y}|\boldsymbol{\theta}) \quad (12)$$

$$\approx \log \pi(\boldsymbol{\theta}) + \log \sum_{s=1}^{S} \pi_{\boldsymbol{\epsilon}}(\mathbf{y}|f(\boldsymbol{\theta}, \omega_s)) - \log(S) \quad (13)$$

we can also maximize the lower-bound of $\mathcal{LJ} \geq \mathcal{LB}$, where

$$\mathcal{LB} \approx \log \pi(\boldsymbol{\theta}) + \sum_{s=1}^{S} \log \pi_{\boldsymbol{\epsilon}}(\mathbf{y}|f(\boldsymbol{\theta}, \omega_s)) \quad (14)$$

which will be more convenient for optimization for typical kernels (i.e. π_ϵ is Gaussian).

The simplest optimization is to set the objective function C to be $-\mathcal{LJ}$ or $-\mathcal{LB}$ and compute stochastic gradients using 2SPSA with q repeats (negative log-likelihood because we will assume minimization):

$$\frac{\partial C}{\partial \boldsymbol{\theta}} = \frac{1}{q}\sum_{j=1}^{q}\frac{C(\boldsymbol{\theta} + c\Delta_j) - C(\boldsymbol{\theta} - c\Delta_j)}{2c\Delta_j} \quad (15)$$

or 1SPSA

$$\frac{\partial C}{\partial \boldsymbol{\theta}} = \frac{1}{q}\sum_{j=1}^{q}\frac{C(\boldsymbol{\theta} + c\Delta_j) - C(\boldsymbol{\theta})}{c\Delta_j} \quad (16)$$

which is less accurate but is less computation if $C(\boldsymbol{\theta})$ is computed at each step (TODO: possible algorithm). TODO: rewrite $\frac{\partial C}{\partial \boldsymbol{\theta}}$ as $\hat{g}$ or similar.

2.2 Special Case: Gaussian Kernels

When the kernel is known we can use the chain rule to compute the gradients with the first part from the kernel and the second from the variation of the simulator pseudo-statistics, whose gradient can be estimated using SPSA. Expanding the likelihood over J statistics, the lower bound on the loglikelihood becomes:

$$\hat{L} = \sum_s \log \pi_\epsilon(\mathbf{y}|f(\boldsymbol{\theta}, \omega_s)) \quad (17)$$

$$= \sum_s \log \prod_{j=1}^{J} \pi_{\epsilon_j}(y_j|f_j(\boldsymbol{\theta}, \omega_s)) \quad (18)$$

$$= \sum_s \sum_j \log \pi_{\epsilon_j}(y_j|f_j(\boldsymbol{\theta}, \omega_s)) \quad (19)$$

$$= \sum_s \sum_j -\frac{1}{2}\frac{(y_j - f_j(\boldsymbol{\theta}, \omega_s))^2}{\epsilon_j^2} - Z_j \quad (20)$$

$$(21)$$

The gradient:

$$\nabla \hat{L} = \sum_s \sum_j \frac{(y_j - f_j(\boldsymbol{\theta}, \omega_s))}{\epsilon_j^2} \cdot \frac{\partial f_j(\boldsymbol{\theta}, \omega_s)}{\partial \boldsymbol{\theta}} \quad (22)$$

$$= \sum_s \sum_j e_{js} \nabla_{js} \quad (23)$$

The first part e_{js} is analytic, the second part ∇_{js} requires estimation by 2SPSA:

$$\hat{\nabla}_{js} = \frac{1}{q}\sum_{r=1}^{q}\frac{f_j(\boldsymbol{\theta} + c\Delta_r, \omega_s) - f_j(\boldsymbol{\theta} - c\Delta_r, \omega_s)}{2c\Delta_r} \quad (24)$$

or 1SPSA

$$\hat{\nabla}_{js} = \frac{1}{q}\sum_{r=1}^{q}\frac{f_j(\boldsymbol{\theta} + c\Delta_r, \omega_s) - f_j(\boldsymbol{\theta}, \omega_s)}{c\Delta_r} \quad (25)$$

the benefit of 1SPSA in this case is that the first gradients e_{js} can be computed from $f_j(\boldsymbol{\theta}, \omega_s)$.

3 Applying SGLD to ABC

There are two main sources of randomness in the ABC gradient computation: the random perturbation mask and the random seed of the simulator model, which controls the simulator noise. This noise can be eliminated by using common random numbers [?]. On the other hand, it is not clear that this is the best strategy, as the noise from the masks at a fixed seed may be higher than

$$
\begin{aligned}
V_{\hat{g}_t^j(\boldsymbol{\theta}_t)} &= \frac{1}{q-1}\sum_{j=1}^{q}(\hat{g}_t^j(\boldsymbol{\theta}_t) - \hat{g}_t(\boldsymbol{\theta}_t))^2 && (26)\\
V_{\hat{g}_t(\boldsymbol{\theta}_t)} &= \frac{1}{q-1} V_{\hat{g}_t^j(\boldsymbol{\theta}_t)} && (27)\\
\frac{\epsilon^2}{4} V_{\hat{g}_t(\boldsymbol{\theta}_t)} &= \frac{\epsilon^2}{4q} V_{\hat{g}_t^j(\boldsymbol{\theta}_t)} && (28)\\
&<< \epsilon && (29)\\
\frac{\epsilon}{4(q-1)} V_{\hat{g}_t^j(\boldsymbol{\theta}_t)} &<< 1 && (30)
\end{aligned}
$$

Check for $g(\boldsymbol{\theta}) + h(\boldsymbol{\theta}) + \eta$ that $h(\boldsymbol{\theta}) + \eta \approx \eta$

Also $\theta_{t+1} = \theta_t + \frac{\epsilon}{2}\hat{g}_t(\boldsymbol{\theta}_t) + \sqrt{\epsilon}\mathcal{N}(0,1)$

3.1 Statistical Tests for Gradients

From Byrd:

$$
\begin{aligned}
\frac{||V_{\hat{g}_t^j(\boldsymbol{\theta}_t)}||_1}{q} &\le r^2 ||\hat{g}_t||_2^2 && (31)\\
\hat{q} &= \frac{||V_{\hat{g}_t^j(\boldsymbol{\theta}_t)}||_1}{r^2||\hat{g}_t||_2^2} && (32)
\end{aligned}
$$

TODO:

- Check that q gradients are normal. Especially as q increases.
- Merge Byrd's test for $\hat{q}$ and condition on SGLD-ABC.
- Variance check for increasing q can also be used to stop optimization since the variance is less than the noise in the gradients.

3.2 SGD, AdaGrad, RMSProp, AdaM

- Arguably the two most useful procedures in simulation-based science are optimization and Bayesian inference.
- Optimization can take the form of simple grid-search to sophisticated techniques like factorial design [?], Bayesian experiment design [?], (mention others).
- Recently, Bayesian optimization techniques have shown success in optimizing very expensive black-box simulators [?].
- Though gradients are not directly computable for simulators, they can be computed analytically by using finite differences (see [?]). This quickly becomes infeasible for large p problems. There is however an alternative stochastic approximation algorithm by Spall [?] that requires only 2 simulation calls independent of p.
- By using this approximation, gradient-based algorithms can be adopted for simulators: for optimization, using analogous stochastic gradient descent algorithms and for Bayesian inference using Langevin dynamics [?].

With our stochastic gradient in hand, we can make use of robust stochastic gradient rules, including recent advances from machine learning.

3.2.1 SGD

The basic update rule is at time t for minimization is

$$
\Delta\boldsymbol{\theta} = -\alpha_t \hat{g}_t \qquad (33)
$$

with conditions $\sum_{t=1}^{\infty}\alpha_t = \infty$ and $\sum_{t=1}^{\infty}\alpha_t^2 < \infty$ required for convergence. An additional momentum parameter β_1 can be added to improve convergence

$$
\begin{aligned}
g_m &= \beta_1 g_m + (1-\beta_1) * \hat{g}_t && (34)\\
\Delta\boldsymbol{\theta} &= -\alpha_t g_m && (35)
\end{aligned}
$$

3.2.2 AdaGrad

AdaGrad uses the total cumulative element-wise square of the gradients to automatically both decrease α_t and scale the learning rates for each dimension:

$$
\begin{aligned}
g_m &= \beta_1 g_m + (1-\beta_1) * \hat{g}_t && (36)\\
v_m &= \sum_{t'=1}^{t} \hat{g}_t{}^2 && (37)\\
\Delta\boldsymbol{\theta} &= -\alpha_t g_m/\sqrt{v_m} && (38)
\end{aligned}
$$

3.2.3 RMSProp

$$
\begin{aligned}
g_m &= \beta_1 g_m + (1-\beta_1) * \hat{g}_t && (39)\\
v_m &= \beta_2 v_m + (1-\beta_2) * \hat{g}_t{}^2 && (40)\\
\Delta\boldsymbol{\theta} &= -\alpha_t g_m/\sqrt{v_m} && (41)
\end{aligned}
$$

3.2.4 AdaM

$$g_m = \beta_1 g_m + (1 - \beta_1) * \hat{g}_t \tag{42}$$

$$v_m = \beta_2 v_m + (1 - \beta_2) * \hat{g}_t{}^2 \tag{43}$$

$$\gamma = \frac{\sqrt{1 - (1 - \beta_2)^t}}{(1 - (1 - \beta_1)^t)} \tag{44}$$

$$\Delta\boldsymbol{\theta} = -\alpha_t \gamma g_m / \sqrt{v_m} \tag{45}$$