

An Error Analysis of Off-Policy LLM RL Updates, Starting from Score-Function Centering

One umbrella formula: three bias terms and one variance formula

October 2026

Contents

Chapter 0	The umbrella formula	2
1	Setting, notation, and the class of estimators	4
2	On-policy: why the baseline does not change the expected update	8
3	Off-policy: deriving the umbrella formula	12
Part I summary: a map of the note and a table of methods		18
4	Term A: what drift is	18
5	Removing term A: making the mean of the multiplier zero	24
6	Term B: the token weights	29
7	Term C: occupancy	32
8	Term D: methods outside Premise P	35
9	Variance: the cost of fixing the bias	40
10	How to choose: target first, then source, then variance	45
11	The TRPO-style bound: which terms it controls and which it does not	49
12	Practical checklist and diagnostics	50
13	Questions and short answers	52
A	Derivations	54

B Optional reading: which training mechanisms may amplify term A	58
C Implementation variants, and where other work fits in the umbrella formula	59
D Notation	62

Chapter 0 The umbrella formula

What this note is about

When a language model is trained with RL, the policy that generates the responses, the **sampler** q , is often not the policy whose parameters are updated, the **trainer** $p = p_\theta$: the sampler may be a separate inference engine, an older copy of the parameters, or a modified sampling distribution (§1.1). Whenever $q \neq p$, the data are off-policy.

At each training step, a batch of responses from the sampler gives an update vector $\hat{g}$; what we want is the gradient $\nabla_\theta J$ of the expected return J . Two questions follow:

1. **On average, does $\hat{g}$ point in the right direction?** The part that does not is the bias; more samples do not remove it.
2. **Can a finite batch see this direction?** The part that hides it is the noise; more samples make it smaller.

Recent methods for these two questions include importance sampling (IS) and its truncated and masked forms, TIS [7] and MIS [8, 9]; score-function centering (SFC) [1, 3]; GSPO [13]; KL penalties toward the sampler [3, 4]; baselines (the group mean [10], leave-one-out [19, 20], length weighting [14], a critic); and loss normalizations (GRPO’s $1/L_i$ [10], DAPO [11], Dr. GRPO [12]). Each claims to fix something. This note writes the error of $\hat{g}$ as one formula and reads each method as removing a particular term of it, at a particular cost.

SFC adds one term to the loss and less than 1% to the wall-clock time; in Marek and Ryabinin’s 30B experiments with the most heavily quantized sampler, it reaches 30% training accuracy, against 12% for TIS and below 5% for every other method [1]. As far as we know, frontier labs already use such a correction in large-scale RL, under names such as “*score function centering*”, “*alternative tokens*”, and “*covariance learning*”. The explanations given so far (distillation toward the sampler [1], the zero-mean score [3], and five readings including an ideal baseline, a straight-through estimator, and a local family of distributions [5]) are all views of one term of the formula below (Chapter 4, Appendix C).

The umbrella formula

One update averages N independent **sampling units**, giving $\hat{g}_N$; a unit is, for example, the group of responses to one prompt, because responses that serve as each other’s baselines are not independent (§9.1). Under Premise P (§1.4), the bias splits exactly into three terms, and the variance is that of

one unit divided by N :

$$\mathbb{E}\|\hat{g}_N - \nabla_{\theta} J\|^2 = \left\| \underbrace{\mathbf{A}}_{\text{drift}} + \underbrace{\mathbf{B}}_{\text{token weights}} + \underbrace{\mathbf{C}}_{\text{occupancy}} \right\|^2 + \frac{\text{tr Cov}(\hat{g})}{N} \quad (1)$$

Each term is a difference left over when we move, one step at a time, from what we compute to what we want (exact definitions in Chapter 3):

- **A, drift.** $\mathbf{A}$ is the part of the signal $R - b$ shared by all tokens at the current prefix (the common offset), times the mean under the sampler of the vector that multiplies the signal (the multiplier). This mean is zero on-policy but generally not off-policy. $\mathbf{A}$ is the only term that depends on the baseline: with a constant reward, $\mathbf{B}$ and $\mathbf{C}$ are zero, but $\mathbf{A}$ generally is not.
- **B, token weights.** Each token’s share of the update is the sampler’s probability times the weight you choose; the target uses the trainer’s probability. $\mathbf{B}$ is the difference.
- **C, prefixes and continuations (occupancy).** Which prefixes occur, and how each is continued, are decided by the sampler, not the trainer. $\mathbf{C}$ is exactly what the TRPO bound [22] controls, at the level of gradients.

Premise P says that the baseline does not look at the current response, and that each token’s weight and centering vector depend only on the current prefix and token. Where a method breaks it (GRPO’s $1/L_i$, the group standard deviation, a length-weighted baseline that includes the response itself, PPO’s sign mask [23]), we take a reference estimator $\hat{g}_0$ that satisfies the premise and call the difference of expected updates **term D**: $\mathbf{D} = \mathbb{E}[\hat{g}] - \mathbb{E}[\hat{g}_0]$. $\mathbf{D}$ depends on this choice (Chapter 8 states it case by case) and exists even on-policy.

The variance also has an exact formula. At one prefix, a single token’s update is $(R - b) f_{\xi}$, where f_{ξ} is the multiplier and ξ is any extra randomness used to compute it (such as extra tokens drawn to estimate an average); its second moment is three kinds of noise times the average size of the multiplier:

$$\mathbb{E}[\|(R - b) f_{\xi}\|^2 | s] = \sum_v q(v | s) \left[\underbrace{(A^q(s, v) + \bar{A}_q(s))^2}_{\text{signal mean}} + \underbrace{\sigma^2(s, v)}_{\text{return noise}} + \underbrace{\text{Var}(b | s)}_{\text{baseline noise}} \right] \mathbb{E}_{\xi} \|f_{\xi}(s, v)\|^2. \quad (2)$$

Three symbols that are easy to confuse. The signal $A = R - b$ of one sample is random (the text mostly writes $R - b$); $A^q(s, v)$ is the true advantage, the expected return when token v is chosen and the sampler finishes the response, minus the value of the prefix; $\bar{A}_q(s) = V^q(s) - \mathbb{E}[b | s]$ is the common offset; and $\mathbb{E}[R - b | s, v] = A^q(s, v) + \bar{A}_q(s)$ (Chapter 1). The upright $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ only label the bias terms and are not the signal A .

The last factor is squared before averaging over ξ : $\mathbb{E}_{\xi} \|f_{\xi}\|^2 = \|\mathbb{E}_{\xi} f_{\xi}\|^2 + \text{tr Cov}_{\xi}(f_{\xi})$, where the second part is the extra variance from estimating an average. So the formula shows why a baseline often lowers the variance (it changes the common offset in the first term), why IS raises it (the multiplier contains the ratio), and why estimating SFC’s center with alternative tokens adds to it (the multiplier becomes random).

How to read this note

- **Core:** Part I, Chapters 1–3 (notation, why nothing goes wrong on-policy, the derivation of the formula), whose map of the note (Figure 1) and table of methods (Table 1) at the end serve as an index for the rest; then Chapters 4–5 (term A and how to remove it) and 9 (the variance).
- **As needed:** Chapters 6–8 (terms B, C, D) and 10–12 (how to choose, a TRPO-style bound, a practical checklist with diagnostics).
- **Chapter 13:** questions and answers by term, each pointing back to the text.
- **Appendices:** derivations (A); optional mechanisms that may amplify term A (B); implementation variants and where other articles fit (C); notation (D).

Chapters 4–9 open with the umbrella formula, their term highlighted, and close with what is left and which cells of Table 1 they fill. The examples: Example 1, a two-token prefix (Chapters 2–9); Example 2, a three-token prefix with a very stale sampler (term B and the variance); Example 3, two solution routes (term C); and several length-weighted baselines (term D).

Main results carry tags: [identity] holds exactly within the stated scope; [model] holds under stated extra assumptions; [empirical] is an experimental result reported in a paper; [toy] is a small simulation described in the text.

Part I Where the umbrella formula comes from

1 Setting, notation, and the class of estimators

This chapter answers: What do the symbols, the “update”, and the “true gradient” mean? Which estimators does the umbrella formula cover?

1.1 Scenario

Take a prompt x . The language model generates a response $y = (y_1, \dots, y_L)$ one token at a time, and its length L can vary. The **prefix** before the t -th token, $s_t = (x, y_{<t})$, is the state, and the t -th token y_t is the action. When the response ends, a verifier or a reward model gives it a number $R = R(x, y)$, the return.

Two versions of the same model are involved:

- **sampler** q : the policy that generates the training data; $q(v \mid s)$ is its probability of choosing token v at prefix s .
- **trainer** $p = p_\theta$: the policy whose parameters θ are being updated.

They differ for three common reasons, which can occur together:

- **engine mismatch:** from the same weights, the inference engine computes slightly different probabilities, for example because of quantization or different kernels;

- **staleness:** in asynchronous training, the data come from the weights of a few updates earlier;
- **sampling settings:** with a temperature other than 1, or top- k or top- p truncation, the sampler uses a modified distribution.

1.2 The objective and the true gradient

The objective is the trainer’s expected return

$$J(\theta) = \mathbb{E}_{y \sim p_\theta(\cdot|x)} [R(x, y)],$$

averaged over prompts; below we look at one prompt.

The score. At a prefix s , the trainer gives each token v in the vocabulary a logit $z(s, v)$, and $p(v | s) = e^{z(s, v)} / \sum_u e^{z(s, u)}$. Let $\nabla_\theta z(s, v)$ be the gradient of this logit with respect to the parameters (a vector as long as θ), and for any distribution π over the vocabulary let

$$m_\pi(s) = \sum_u \pi(u | s) \nabla_\theta z(s, u)$$

be its average under π . Differentiating $\log p(v | s)$ (Appendix A.1) gives the trainer’s **score**:

$$g(s, v) = \nabla_\theta \log p(v | s) = \nabla_\theta z(s, v) - m_p(s). \quad (3)$$

In one sentence: To raise the probability of token v , its logit has to go up relative to the trainer’s own average logit, because all the probabilities must add up to 1.

Averaged under the trainer’s own distribution, the score is zero, the most basic zero in this note:

$$\mathbb{E}_{v \sim p(\cdot|s)} [g(s, v)] = m_p(s) - m_p(s) = 0. \quad (4)$$

Values and advantages. The superscript says whose continuation is used. For a policy $\pi \in \{q, p\}$, let $Q^\pi(s, v)$ be the expected return when token v is chosen at prefix s and π then finishes the response, let $V^\pi(s) = \sum_v \pi(v | s) Q^\pi(s, v)$ be the value of the prefix, and let

$$A^\pi(s, v) = Q^\pi(s, v) - V^\pi(s)$$

be the true advantage, which averages to zero under π by definition: $\sum_v \pi(v | s) A^\pi(s, v) = 0$. The data are generated by q , so the continuation values observed in the data are Q^q , not Q^p (this matters in Chapter 7).

The true gradient. The policy gradient theorem says (Appendix A.2)

$$\nabla_\theta J = \sum_t \mathbb{E}_{s_t \sim p} \left[\sum_v p(v | s_t) A^p(s_t, v) g(s_t, v) \right]. \quad (5)$$

Here $\mathbb{E}_{s_t \sim p}$ averages over the t -th prefix when the trainer generates the response (a response that has ended before step t contributes nothing), and $\sum_t$ sums over positions. The advantage is the trainer’s own A^p , because the trainer also generates the continuation.

For a prefix s and any nonnegative token weights $\mu(v)$, define

$$G_s[\mu] = \sum_v \mu(v) A^q(s, v) g(s, v), \quad G_s^p[\mu] = \sum_v \mu(v) A^p(s, v) g(s, v). \quad (6)$$

The two differ only in whose continuation the advantage uses, and $\nabla_\theta J = \sum_t \mathbb{E}_{s_t \sim p} [G_{s_t}^p[p]]$.

In one sentence: $G_s[\mu]$ has, for each token v , a component along $g(v)$ scaled by $\mu(v)A^q(v)$, which raises the probability of v when $A^q(v) > 0$ and lowers it when $A^q(v) < 0$; the weights μ need not add up to 1.

1.3 The reference target: which policy we want to improve

Bias is measured against a reference target. The default is $\nabla_\theta J(p_\theta)$, the gradient of the current trainer; under staleness it is always this one, because the trainer is what gets deployed. Under engine mismatch, if this inference engine is what is deployed, the policy to improve is the sampler q_θ tied to the current parameters, and the target is $\nabla_\theta J(q_\theta)$. Chapter 3 writes the umbrella formula for both.

1.4 The class of estimators and Premise P

Every method in this note computes, from one response, an update of the same form:

$$\hat{g} = \sum_{t=1}^L (R - b) f_t, \quad f_t = w(s_t, y_t) g(s_t, y_t) - c(s_t) \quad (7)$$

There are three things to choose:

- **baseline** b : the number subtracted from the return. It can be a constant, be computed from other responses to the same prompt (leave-one-out, group mean), or depend on the prefix (a critic), in which case it is really $b_t = b(s_t)$; Eq. (7) and the rest of the note just write b .
- **weight** $w(s, v)$: the scalar this token is multiplied by. Plain REINFORCE [15] uses $w = 1$; exact IS uses the ratio $r = p(v | s)/q(v | s)$; truncated IS uses $\tilde{w} = \min(r, \bar{r})$, where $\bar{r} \geq 1$ is the ratio cap (not the centering vector c below), or sets the weight to zero for tokens whose ratio falls outside an interval.
- **centering vector** $c(s)$: a vector that depends only on the prefix and is subtracted from the multiplier. Plain REINFORCE and IS use $c = 0$; SFC [1] uses $c = \mathbb{E}_q[g | s]$; truncation with re-centering uses $c = \mathbb{E}_q[\tilde{w}g | s]$.

We call f_t , the vector that multiplies the signal $R - b$, the **multiplier**.

Premise P. The umbrella formula requires three conditions:

- (P1) Given the current prefix s_t , b is independent of the rest of this response (the current token and the continuation) and of the extra random numbers. A constant, a baseline computed from other responses, and a critic that depends only on the prefix satisfy this; a baseline that includes this response’s own return does not.

- (P2) w depends only on s_t , the current token y_t , and extra random numbers ξ that are independent of this response given the prefix.
- (P3) c depends only on s_t and such ξ (for example, the few extra tokens that SFC samples at this prefix to estimate the average).

Estimators that satisfy (P1)–(P3) are **token-local estimators**. (P1) is the reason for the leave-one-out baseline (Chapter 2); (P2) and (P3) mean that the multiplier may not look at the later tokens of this response or at the return. Real methods violate these conditions in a few places, treated in Chapters 8 and 7:

- GRPO’s $1/L_i$ [10] depends on the length of the whole response, which violates (P2);
- the group mean includes this response’s own return in the baseline, which violates (P1), and the group standard deviation makes the scale of the signal depend on that return too;
- a sequence-level product of ratios depends on the whole response;
- PPO-style clipping [23] makes the weight depend on the sign of the signal, which also violates (P2).

The multiplier actually used, and its mean. With extra random numbers, the multiplier actually used and its average over them are

$$f_\xi(s, v) = w_\xi(s, v) g(s, v) - c_\xi(s), \quad f_0(s, v) = \mathbb{E}_\xi[f_\xi(s, v)].$$

The expected update needs only f_0 ; the variance needs f_ξ itself, because the mean of a square is not the square of the mean (§3.8). Without extra random numbers (for example, when m_q is computed exactly), the two are the same and we write f .

Three quantities at each prefix. At a prefix s , (w, c) and b define:

$$\begin{aligned} \mu(v \mid s) &= q(v \mid s) \mathbb{E}_\xi[w_\xi(s, v)] && \text{(weight of each token in the update)} \\ \bar{f}(s) &= \mathbb{E}_{v \sim q}[f_0(s, v)] && \text{(mean of the multiplier)} \\ \bar{A}_q(s) &= V^q(s) - \mathbb{E}[b \mid s] && \text{(common offset)} \end{aligned} \tag{8}$$

The **common offset** $\bar{A}_q$ is the same for every token at this prefix, so it does not help distinguish the tokens. A baseline computed from other responses does not depend on the prefix, so $\mathbb{E}[b \mid s] = \mathbb{E}[b]$; a critic’s $b(s)$ is a function of the prefix, so $\mathbb{E}[b \mid s] = b(s)$.

1.5 Notation conventions

- The signal $A = R - b$ of one sample is random; given the prefix and the token, $\mathbb{E}[A \mid s, v] = Q^q(s, v) - \mathbb{E}[b \mid s] = A^q(s, v) + \bar{A}_q(s)$.
- For a single prefix we omit s and write $q(v)$, $g(v)$, $A^q(v)$, $\bar{A}_q$.
- Updates are vectors: “variance” means $\text{tr Cov}(\hat{g})$, the sum of the variances of the components, and the squared bias is the squared length of the bias vector.
- n is the number of responses in a group, L_i is the length of the i -th response, and s_{it} is the prefix before the t -th token of the i -th response.

Appendix D lists all the notation.

Next question: We start with the simplest case, $q = p$: why does subtracting any baseline leave the expected update unchanged?

2 On-policy: why the baseline does not change the expected update

This chapter answers: When the trainer generates the data itself ($q = p$), why does a baseline leave the expected update unchanged, and when does this fail? What does the baseline change instead?

This whole chapter is on-policy: $q = p$, $w = 1$, and $c = 0$, so the multiplier is the trainer’s score, $f_t = g(s_t, y_t)$.

2.1 An identity at one prefix

At a prefix s , sample a token v from p , compute the signal $A = R - b$, and update with $A g(v)$: this is REINFORCE with a baseline [15]. If b is a fixed number, the expected update is

$$\mathbb{E}_p[(R - b)g] = \mathbb{E}_p[Rg] - b\mathbb{E}_p[g] = \mathbb{E}_p[Rg], \quad (9)$$

where the second step uses Eq. (4), so it does not depend on b . Given v , the conditional mean of R is $Q^p(v)$, so $\mathbb{E}_p[Rg] = \sum_v p(v)Q^p(v)g(v) = \sum_v p(v)A^p(v)g(v) + V^p \sum_v p(v)g(v) = G_s^p[p]$, which is exactly the true-gradient term at this prefix.

In one sentence: The baseline leaves the expected update unchanged only because the score averages to zero under the distribution the token is sampled from.

Example 1 (on-policy): two tokens

The model has generated a fairly good prefix, and there are two possible next tokens. The success rate after choosing token 1 and finishing the response is $Q(1) = 0.91$; after token 2 it is $Q(2) = 0.89$. The trainer picks token 1 with probability $p(1) = 0.65$, so the value of the prefix is $V^p = 0.65 \times 0.91 + 0.35 \times 0.89 = 0.903$ and the true advantage is $A^p = (0.007, -0.013)$.

The only parameter θ is the trainer’s **log-odds** of picking token 1: the logit of token 1 is θ and that of token 2 is fixed at 0, so $p(1) = e^\theta / (1 + e^\theta)$, $\nabla_\theta z(1) = 1$, $\nabla_\theta z(2) = 0$, $m_p = p(1) = 0.65$, and the score is

$$g(1) = 1 - 0.65 = 0.35, \quad g(2) = 0 - 0.65 = -0.65, \quad \mathbb{E}_p[g] = 0.65 \times 0.35 - 0.35 \times 0.65 = 0.$$

The true gradient is $\sum_v p(v)A^p(v)g(v) = 0.65 \times 0.007 \times 0.35 + 0.35 \times (-0.013) \times (-0.65) = 0.00455$. For any b , the expected update is

$$0.65(0.91 - b)(0.35) + 0.35(0.89 - b)(-0.65) = 0.00455,$$

because the coefficient of b , $-0.65 \times 0.35 + 0.35 \times 0.65$, is exactly zero.

2.2 When the baseline is random: Premise (P1) and three zeros

In practice the baseline is estimated from the data, for example as the average return of the other responses to the same prompt, so the second step of Eq. (9) has to change. Let $S = \sum_t g(s_t, y_t)$ be the sum of a response’s scores; it is $\nabla_\theta \log p(y \mid x)$, because the probability of a response is the product of its tokens’ probabilities. If the i -th response has a baseline b_i , then (Appendix A.2)

$$\mathbb{E}[(R_i - b_i) S_i] = \nabla_\theta J - \mathbb{E}[b_i S_i]. \quad (10)$$

So the baseline leaves the expectation unchanged when $\mathbb{E}[b_i S_i] = 0$. The simplest guarantee is Premise (P1), b_i independent of the i -th response: then $\mathbb{E}[b_i S_i] = \mathbb{E}[b_i] \mathbb{E}[S_i] = \mathbb{E}[b_i] \times 0 = 0$. This uses two facts: independence splits the expectation into a product, and the score has mean zero. Off-policy the first is still assumed, but the second fails; this is term A (Chapter 4).

Leave-one-out baseline. Sample n responses to the same prompt, and for the i -th response use the average of the other $n - 1$:

$$b_{-i} = \frac{1}{n-1} \sum_{j \neq i} R_j. \quad (11)$$

It does not look at the i -th response, so it satisfies (P1), and its mean is the value at the prompt, $V^P(s_0)$. REINFORCE with this baseline is RLOO [19, 20]. One response is left out for independence, not for accuracy (the average of $n - 1$ is slightly noisier than the average of n).

Three statements of the form “something equals zero” are easy to mix up:

Statement	What it means	Does an unbiased update need it?
(a) $\sum_v \pi(v) A^\pi(v) = 0$	The true advantage averages to zero under its own policy	No. It holds by definition. In Chapter 3 it makes the centering vector c disappear from the signal term
(b) $\mathbb{E}[b_i S_i] = 0$	The baseline contributes nothing to the expected update	Yes. On-policy, it holds as long as b_i does not depend on the i -th response
(c) In a batch, $\sum_i A_i = 0$ or $\sum_i L_i A_i = 0$	The total positive signal equals the total negative signal, per response or per token	No. A property of one batch, useful only in the analysis of the variance and of the off-policy case (Chapter 4)

Two small examples show that (a) is not the right criterion. With $b_i = R_i$, every response’s “advantage” is zero, so it certainly averages to zero, but the update is zero too and the gradient is lost; (b) fails, because $\mathbb{E}[b_i S_i] = \mathbb{E}[R_i S_i] = \nabla_\theta J \neq 0$. With $b = 100$, the advantage $R - 100$ has mean $J - 100$, far from zero, yet the update is unbiased, because a constant satisfies (b); it is just very noisy.

Self-inclusion rule. GRPO’s baseline [10] is the group mean $\bar{R} = \frac{1}{n} \sum_j R_j$, which includes the i -th response’s own reward, so it does not satisfy (P1). In general, let the baseline give the response’s own reward a weight ω_i (a scalar, unrelated to the token weight w) and the rest of the weight to quantities that do not involve this response: $b_i = \omega_i R_i + (1 - \omega_i) b_{-i}$. Then

$$R_i - b_i = (1 - \omega_i) (R_i - b_{-i}). \quad (12)$$

So including a response in its own baseline scales its signal by $1 - \omega_i$:

- $\omega_i = 0$ is RLOO, which is unbiased;
- $\omega_i = 1/n$ for every response is GRPO’s group mean: the expected update is $(1 - 1/n)\nabla_{\theta}J$, a harmless rescaling;
- $\omega_i = 1$ is the example $b_i = R_i$ above: the signal is gone;
- when ω_i varies across responses, they are scaled by different factors, so the **direction** of the update changes; this is the length bias of the length-weighted baseline (Chapter 8).

2.3 Splitting the signal into four terms

For the t -th token, the signal $R - b$ splits exactly into four terms (for the advantage in the form $Q - V$, see [16]):

$$R - b = \underbrace{Q^q(s_t, y_t) - V^q(s_t)}_{\text{(i) true advantage}} + \underbrace{R - Q^q(s_t, y_t)}_{\text{(ii) return noise}} + \underbrace{V^q(s_t) - V^q(s_0)}_{\text{(iii) prefix offset}} + \underbrace{V^q(s_0) - b}_{\text{(iv) baseline error}} \quad (13)$$

The values are computed under the policy that generated the data, so they are written with q ; in this chapter $q = p$.

- (i) is what we want.
- (ii) has mean zero once the prefix and the token are fixed, so it only adds noise, on-policy and off-policy alike.
- (iii) depends only on the prefix, not on the token chosen at step t ; (iv) is the same number for every token of the response, as long as b does not look at this response. On-policy, both multiply a score with mean zero, so they also only add noise.

Given the prefix, the mean of (iii) plus (iv) is the common offset $\bar{A}_q(s_t) = V^q(s_t) - \mathbb{E}[b \mid s_t]$. We keep the two apart because a prompt-level baseline (such as the leave-one-out average) can remove (iv) in expectation, but not (iii), which changes with the prefix.

2.4 What the baseline changes is the variance

Take the simplest case: one prefix, a deterministic return ($R = Q^p(v)$), and a fixed b . The single-sample update $X = (Q^p(v) - b)g(v)$ has the true gradient $G_s^p[p]$ as its mean, and its variance is

$$\text{Var}(X) = \sum_v p(v) (Q^p(v) - b)^2 \|g(v)\|^2 - \|G_s^p[p]\|^2.$$

Averaging over N independent samples divides the variance by N (on what counts as independent, see §9.1). A useful quantity is the **number of samples needed for the signal to be as large as the noise**:

$$N^* = \frac{\text{Var}(X)}{\|\mathbb{E}X\|^2}. \quad (14)$$

With N samples, the averaged update has standard deviation $\sqrt{\text{Var}(X)/N}$ and mean $\|\mathbb{E}X\|$, which are equal at $N = N^*$. In one dimension, with roughly normal noise, the average of N samples has the

sign of the true gradient with probability $\Phi(\sqrt{N/N^*})$, where Φ is the standard normal cumulative distribution function: about 69% at $N = N^*/4$, 84% at $N = N^*$, and 98% at $N = 4N^*$.

Example 1 (continued): the baseline and N^* (deterministic return, fixed baseline)

We keep the two-token prefix and treat Q as the deterministic return after the chosen token, not the random outcome of one success or failure (random returns: §9.3). The single-sample update X is $(0.91 - b) \times 0.35$ for token 1 (probability 0.65) and $(0.89 - b) \times (-0.65)$ for token 2 (probability 0.35).

baseline b	0	0.5 (prompt's success rate)	$0.903 = V^p$	$0.897 = b_{\text{var}}$
X for token 1	0.3185	0.1435	0.0025	0.0045
X for token 2	-0.5785	-0.2535	0.0085	0.0045
mean	0.00455	0.00455	0.00455	0.00455
variance	0.183	0.0359	8×10^{-6}	0
N^*	8842	1732	0.4	0

At $b = 0.5$, every sample's signal contains a common part of about 0.4 (success rate about 0.9, baseline 0.5); multiplied by scores of opposite signs, it produces large fluctuations. At $b = V^p$ only $A^p = (0.007, -0.013)$ is left, both possible updates are positive, and one sample already has the right sign.

The baseline that minimizes the variance. In this setting, the b that minimizes $\mathbb{E}[(Q^p - b)^2 \|g\|^2]$ is [17, 18]

$$b_{\text{var}} = \frac{\mathbb{E}_p[R \|g\|^2]}{\mathbb{E}_p[\|g\|^2]}, \quad (15)$$

the average return with each outcome weighted by the squared size of its score. It equals V^p if and only if $\|g\|^2$ is uncorrelated with the return. In Example 1 the scores differ in size ($|g| = (0.35, 0.65)$), and weighting by $p \|g\|^2$ gives the two outcomes weights 0.35 and 0.65, the reverse of p . Still, $b_{\text{var}} = 0.897$ is close to $V^p = 0.903$, because the two values of Q differ by only 0.02, so any weighted average lies in $[0.89, 0.91]$. Its zero variance in the table is special to two tokens and one parameter, where one baseline can make the two values equal.

The value baseline is not always close to the minimum-variance baseline. For example, take $p = (0.99, 0.01)$, returns $R = (1, 0)$, and the log-odds of the first action as the parameter, so $g = (0.01, -0.99)$. Then $V^p = 0.99$ but $b_{\text{var}} = 0.01$. All three baselines give the expected update 0.0099, but the single-sample variance is about 1.0×10^{-6} for $b = 0$, about 9.5×10^{-3} for $b = V^p$, and zero for $b = b_{\text{var}}$. The reason is that the rare bad action has a large score; after subtracting 0.99 it still carries a large negative signal, and the product is a rare, large positive update. So “a critic reduces variance” is the common case, not a guarantee.

In practice, b_{var} is not available: it requires the correlation between the return and the size of the score at every prefix. What can be used is an estimate of V (a critic) or a prompt-level group mean. Chapter 9 shows that with SFC, a baseline that differs from b_{var} adds only variance, no bias, and the added amount has an exact formula.

Next question: The next chapter uses data from the sampler: the vector that multiplies (iii) and (iv) no longer has mean zero, so they become bias (Chapter 4 is about them).

3 Off-policy: deriving the umbrella formula

This chapter answers: When the data come from a sampler q , what is the expected update? In which terms does it differ from the true gradient $\nabla_{\theta} J$, and which knob controls each term?

3.1 The zero mean no longer holds

We average the trainer’s score under the sampler’s distribution:

$$\mathbb{E}_{v \sim q}[g(s, v)] = m_q(s) - m_p(s) = -\nabla_{\theta} D_{\text{KL}}(q(\cdot | s) \| p_{\theta}(\cdot | s)). \quad (16)$$

The second equality treats q as a fixed distribution (Appendix A.3): $\sum_v q(v)g(v) = \sum_v q(v)\nabla_{\theta} \log p_{\theta}(v)$, and $-\sum_v q(v)\log p_{\theta}(v)$ is $D_{\text{KL}}(q \| p_{\theta})$ plus a constant that does not depend on θ . So this average points in the direction that makes p_{θ} more like q .

It is zero when $q = p$ and in general nonzero when $q \neq p$: when the parameters are the logits themselves ($\nabla_{\theta} z(v) = e_v$, where e_v is the one-hot vector of token v), it is exactly $q - p$. With shared parameters, however, it can be zero by chance. With logits $z = (\theta, \theta, 0)$ and $\theta = \log(1/2)$, we get $p = (1/4, 1/4, 1/2)$; take $q = (0.4, 0.1, 0.5)$. Because $\nabla_{\theta} z = (1, 1, 0)$, the average logit gradient is $1/2$ under both, so $q \neq p$ and yet $\mathbb{E}_q[g] = 0$.

Example 1 (off-policy): the same prefix, with data from a stale sampler

The prefix is that of Chapter 2: $Q^q(1) = 0.91$, $Q^q(2) = 0.89$, and the trainer has $p(1) = 0.65$. The data now come from a sampler a few steps old, with $q(1) = 0.6$. The KL between the two is small:

$$D_{\text{KL}}(q \| p) = 0.6 \log \frac{0.6}{0.65} + 0.4 \log \frac{0.4}{0.35} \approx 0.0054.$$

The score is unchanged, $g = (0.35, -0.65)$, but its average under q is $0.6 \times 0.35 - 0.4 \times 0.65 = -0.05$, which is exactly $q(1) - p(1)$. Running REINFORCE directly on the sampler’s data (from here on, **plain REINFORCE**) gives the expected update

$$0.6(0.91 - b)(0.35) + 0.4(0.89 - b)(-0.65) = -0.0403 + 0.05b.$$

For $b < 0.806$ it is negative, so it lowers the log-odds of token 1, the better token; on-policy, the same update is $+0.00455$ for every b (Chapter 2).

In one sentence: The gradient of the expected return does not depend on where the zero of the reward is, so an update whose direction changes with b does not estimate this gradient.

3.2 The five-step derivation

Under Premise P, we take the expectation of the token-local estimator $\hat{g} = \sum_t (R - b) f_t$ of §1.4, jointly over the response generated by the sampler, the baseline, and the extra random numbers.

Step 1: separate out the baseline. For each t , we first condition on the prefix s_t . By (P1), given s_t , the baseline b is independent of the current token, the continuation, and the extra random numbers, so

$$\mathbb{E}[(R - b) f_t \mid s_t] = \mathbb{E}[R f_t \mid s_t] - \mathbb{E}[b \mid s_t] \mathbb{E}[f_t \mid s_t].$$

Off-policy, $\mathbb{E}[f_t \mid s_t]$ is in general not zero, and the subtracted term is the source of term A. So the baseline enters the expected update only through its **conditional mean** $\mathbb{E}[b \mid s_t]$ (§1.4); its fluctuations enter only the variance.

Step 2: take the conditional expectation over the continuation. Given the prefix s_t and the token $y_t = v$, the conditional mean of the return is $Q^q(s_t, v)$, because the sampler generates the continuation. By (P2) and (P3), f_t then depends only on extra random numbers that are independent of the continuation, so $\mathbb{E}[R f_t \mid s_t, v] = Q^q(s_t, v) f_0(s_t, v)$. Averaging over v ,

$$\mathbb{E}[R f_t \mid s_t] = \sum_v q(v \mid s_t) Q^q(s_t, v) f_0(s_t, v),$$

where $f_0(s, v) = \mathbb{E}_\xi[f_\xi(s, v)]$ (§1.4; for SFC, the exactly centered multiplier).

Step 3: split Q^q into advantage plus value. With $Q^q(s, v) = A^q(s, v) + V^q(s)$, the advantage part is

$$\sum_v q(v) A^q(v) f_0(v) = \sum_v q(v) \mathbb{E}_\xi[w_\xi(v)] A^q(v) g(v) - \mathbb{E}_\xi[c_\xi] \sum_v q(v) A^q(v) = G_s[\mu],$$

by zero (a): $\sum_v q(v) A^q(v) = 0$. So the centering vector c drops out of the signal term and affects only the mean of the multiplier. The value part is $V^q(s) \sum_v q(v) f_0(v) = V^q(s) \bar{f}(s)$.

Step 4: put the pieces together. The term subtracted in Step 1 is $\mathbb{E}[b \mid s_t] \bar{f}(s_t)$. Combining it with the $V^q(s_t) \bar{f}(s_t)$ of Step 3 and averaging over the prefix ($\mathbb{E}_{s_t \sim q}$ averages over the distribution of the t -th prefix under the sampler) gives

$$\boxed{\mathbb{E}[\hat{g}] = \sum_t \mathbb{E}_{s_t \sim q} \left[G_{s_t}[\mu] + \bar{A}_q(s_t) \bar{f}(s_t) \right]} \quad (17)$$

where $\bar{A}_q(s_t) = V^q(s_t) - \mathbb{E}[b \mid s_t]$. The first term is the **signal term**, and the second is the **coupling term**, which we call drift. At one prefix, the formula can also be written in covariance form: $\mathbb{E}_q[A f \mid s] = \text{Cov}_q(A, f \mid s) + \mathbb{E}_q[A \mid s] \bar{f}(s)$, where $\text{Cov}_q(A, f \mid s) = G_s[\mu]$ and $\mathbb{E}_q[A \mid s] = \bar{A}_q(s)$.

Step 5: compare with the on-policy case. When $q = p$, $w = 1$, and $c = 0$, we have $\bar{f} = \mathbb{E}_p[g] = 0$ (Eq. (4)), $\mu = p$, prefixes distributed as under p , and $A^q = A^p$, so Eq. (17) is exactly Eq. (5). Off-policy, each of these four equalities can fail, and what the failures leave are the three terms below (the third in two steps).

In one sentence: The expected update is an advantage update at each prefix with weights μ , plus the common offset times the mean of the multiplier.

3.3 The three bias terms

Starting from Eq. (17), we remove the term with the mean of the multiplier, then change the weights, the prefix distribution, and the continuation to those of p , in that order, which gives the true gradient $\nabla_\theta J = \sum_t \mathbb{E}_{s_t \sim p}[G_{s_t}^p[p]]$. Each change leaves one difference term:

$$\boxed{\mathbb{E}[\hat{g}] - \nabla_\theta J = A + B + C} \quad (18)$$

$$A = \sum_t \mathbb{E}_{s_t \sim q}[\bar{A}_q(s_t) \bar{f}(s_t)] \quad (\text{drift: common offset times the mean of the multiplier}) \quad (19)$$

$$B = \sum_t \mathbb{E}_{s_t \sim q}[G_{s_t}[\mu] - G_{s_t}[p]] \quad (\text{token weights: } \mu \text{ instead of } p) \quad (20)$$

$$C = \sum_t \mathbb{E}_{s_t \sim q}[G_{s_t}[p]] - \nabla_\theta J \quad (\text{occupancy: the sampler's prefixes and continuations}) \quad (21)$$

C can be split further into two steps:

$$C = \underbrace{\sum_t (\mathbb{E}_{s_t \sim q} - \mathbb{E}_{s_t \sim p})[G_{s_t}[p]]}_{\text{C1: prefix distribution}} + \underbrace{\sum_t \mathbb{E}_{s_t \sim p}[G_{s_t}[p] - G_{s_t}^p[p]]}_{\text{C2: continuation (A}^q \text{ replaced by A}^p)}.$$

The meaning of the three terms:

- **A is a number times a direction:** the common offset $\bar{A}_q(s_t)$ times the mean of the multiplier $\bar{f}(s_t)$. If either factor is zero, A is zero. This is the most important term in the note (Chapters 4 and 5).
- **B depends only on the weight w :** it is $\sum_t \mathbb{E}_q[G[q] - G[p]]$ when $\mu = q$ (plain REINFORCE, SFC), and zero when $\mu = p$ (exact IS; Chapter 6).
- **C is the gradient of the difference between the TRPO surrogate and the objective:** the surrogate of TRPO [22] and CPI [21] is $L_q(\pi) = J(q) + \sum_t \mathbb{E}_{s_t \sim q}[\sum_v \pi(v | s_t) A^q(s_t, v)]$, and its gradient with respect to θ is exactly $\sum_t \mathbb{E}_{s_t \sim q}[G_{s_t}[p]]$, so $C = \nabla_\theta [L_q(p_\theta) - J(p_\theta)]$. Chapter 7 covers the two sources of C, and Chapter 11 how the TRPO bound limits it.

The order of the changes is our choice: if we change the continuation before the prefix distribution, the forms of C1 and C2 change, but A, B, and the total C do not.

Beyond Premise P: term D. When Premise P does not hold, Step 1 or Step 2 fails, and Eq. (17) cannot be used directly. We then replace the part that violates P with a version that satisfies it (for example, a baseline that includes the response itself becomes a leave-one-out baseline, and $1/L_i$ becomes a constant divisor), compute A, B, and C for the resulting reference estimator $\hat{g}_0$, and define

$$D = \mathbb{E}[\hat{g}] - \mathbb{E}[\hat{g}_0],$$

so that $\mathbb{E}[\hat{g}] - \nabla_\theta J = A + B + C + D$. D depends on how $\hat{g}_0$ is chosen, and Chapter 8 states the choice for each case. In one case D can be written down directly: when the baseline depends on the current response but the multiplier still satisfies (P2) and (P3) (with $\bar{A}_q$ defined as before),

$$\mathbb{E}[(R - b)f | s] = G_s[\mu] + \bar{A}_q(s) \bar{f}(s) - \text{Cov}(b, f | s), \quad (22)$$

and the last term is D (with a reference estimator whose baseline is independent of this response and has the same conditional mean $\mathbb{E}[b \mid s]$). The derivation takes one step, $\mathbb{E}[bf \mid s] = \mathbb{E}[b \mid s] \bar{f}(s) + \text{Cov}(b, f \mid s)$; the rest is as in the five steps. So even when $\bar{f}$ is zero, a baseline that depends on the current response can still change the expected update.

3.4 Which term each of the three knobs controls

We plug the multipliers discussed in this note into $\bar{f} = \mathbb{E}_q[wg] - c$ and $\mu = qw$:

Method	w	c	Mean of the multiplier $\bar{f}$	Weight μ
Plain REINFORCE	1	0	$m_q - m_p = -\nabla_{\theta} D_{\text{KL}}(q \parallel p)$	q
SFC	1	$m_q - m_p$	0	q
Exact IS	$r = p/q$	0	$\mathbb{E}_p[g] = 0$ (needs support coverage)	p
TIS/MIS, no re-centering	$\tilde{w}$	0	$\mathbb{E}_q[\tilde{w}g] \neq 0$	$\min(p, \bar{r}q)$, or masked p
TIS/MIS with re-centering	$\tilde{w}$	$\mathbb{E}_q[\tilde{w}g]$	0	same as above
Critic as baseline	unchanged	unchanged	unchanged	unchanged; changes $\bar{A}_q$

So b controls $\bar{A}_q$ and c controls $\bar{f}$, and together they control A; w controls μ , and therefore B; C has no token-level knob.

3.5 Two reference targets

Eq. (18) takes the gradient of the current trainer, $\nabla_{\theta} J(p_{\theta})$, as the reference target. If the deployed model is this inference engine itself (§1.3), the sampler is $q_{\theta} = \text{softmax}(z_{\theta} + \varepsilon)$, tied to the current parameters, where ε is the engine’s logit error. Under the **fixed-error model** (ε does not change near θ , which amounts to a stop-gradient on the error), the sampler’s own score is

$$\nabla_{\theta} \log q_{\theta}(v \mid s) = \nabla_{\theta} z(s, v) - m_q(s) = h(s, v),$$

which is exactly the SFC score of Chapter 5 (Appendix A.5). So the sampler’s own gradient is

$$\nabla_{\theta} J(q_{\theta}) = \sum_t \mathbb{E}_{s_t \sim q} \left[\sum_v q(v \mid s_t) A^q(s_t, v) h(s_t, v) \right] = \sum_t \mathbb{E}_{s_t \sim q} [G_{s_t}[q]],$$

where the second equality holds because h and g differ only by a vector that does not depend on v , and $\sum_v q A^q = 0$. Subtracting this from Eq. (17) gives

$$\mathbb{E}[\hat{g}] - \nabla_{\theta} J(q_{\theta}) = A + \sum_t \mathbb{E}_{s_t \sim q} [G_{s_t}[\mu] - G_{s_t}[q]].$$

C does not exist, because the prefixes and continuations already come from q . A method with $\mu = q$ (SFC) is left with only A, and A is zero for SFC, so under this model the expected update of SFC

is exactly the reference target. Conversely, using ratios to reweight toward the trainer turns the expected update into the gradient of a policy that will not be deployed. This conclusion needs three conditions at once: the fixed-error model, deployment on this same engine, and accurate centering (Chapter 10 discusses when they hold).

3.6 Numerical check of Example 1

Example 1 has one prefix and the same continuation on both sides, so C is zero and only A and B need checking. The data are those of Example 1, plus $V^q = 0.902$ and $A^q = (0.008, -0.012)$.

- Signal term: $G[q] = 0.6 \times 0.008 \times 0.35 + 0.4 \times (-0.012) \times (-0.65) = 0.0048$.
- The term of the true gradient at this prefix is $G^p[p] = \sum_v p(v) A^p(v) g(v)$. Here $Q^p = Q^q$, but $V^p = 0.903 \neq V^q = 0.902$, so $A^p = (0.007, -0.013)$ and A^q differ by a constant. This constant multiplies $\sum_v p(v) g(v) = 0$, so $G^p[p] = G[p] = 0.65 \times 0.008 \times 0.35 + 0.35 \times (-0.012) \times (-0.65) = 0.00455$, the true gradient of Chapter 2.
- $B = G[q] - G[p] = 0.00025$.

At $b = 0.5$, the expected update of plain REINFORCE is $G[q] + A = 0.0048 - 0.0201 = -0.0153$, which matches $-0.0403 + 0.05 \times 0.5$ from Example 1; its difference from the true gradient is $-0.0153 - 0.00455 = -0.0199 = A + B$.

Baseline b	0	0.5	$0.902 = V^q$	1
Common offset $\bar{A}_q$	0.902	0.402	0	-0.098
$A = \bar{A}_q \times (-0.05)$	-0.0451	-0.0201	0	+0.0049
Signal term $G[q]$	0.0048	0.0048	0.0048	0.0048
Plain REINFORCE (sum of the two)	-0.0403	-0.0153	+0.0048	+0.0097

At $b = 0.5$, A is 4.2 times as large as the signal term and has the opposite sign, so the update points the wrong way. At $b = V^q$, plain REINFORCE gives 0.0048, the same as SFC later, which is still 0.00025 away from the true gradient 0.00455: that difference is B, and it does not depend on the baseline.

3.7 A is a different kind of term from B and C

For weight rules that recover the on-policy update when $q = p$ ($w = 1$, $w = r$, and truncated ratios all do), all three terms go to zero with $p - q$, and they are first-order quantities:

- $m_q - m_p = \sum_v (q - p) \nabla_{\theta} z$;
- $G[q] - G[p] = \sum_v (q - p) A^q g$;
- $L_q(\pi)$ and $J(\pi)$ agree to first order at $\pi = q$, with a difference controlled by the square of the total variation distance (Chapter 11).

General token-dependent weights change A and B even on-policy, so they are not covered by this statement.

The difference lies in the other factor. B and C contain the true advantage A^q , that is, the differences in quality between tokens; A contains the common offset $\bar{A}_q$, the difference between the baseline and the value of the prefix. So when the task is nearly learned and A^q becomes small, B and C become small with it, but A does not.

For example, take the reward to be a constant. Then $A^q \equiv 0$, and B and C are both zero (there is nothing to learn, and nothing should move), but $A = \sum_t \mathbb{E}_q[(R - \mathbb{E}[b | s_t])\bar{f}(s_t)]$ is in general not zero when the baseline differs from that constant (unless the mean of the multiplier is zero or the terms at different prefixes cancel exactly). In Example 1, replace Q^q with $(0.7, 0.7)$ and take $b = 0.5$: the expected update of plain REINFORCE is -0.01 , while for SFC and for exact IS it is zero.

3.8 The variance side

The second part of the mean squared error is $\text{tr Cov}(\hat{g})/N$, where N is the number of independent sampling units (for the correlation between responses in the same group, see §9.1). At one prefix, the update from a single token is $(R - b)f_\xi$. Given (s, v) , we split the signal into three mutually independent parts:

$$R - b = \underbrace{A^q(s, v) + \bar{A}_q(s)}_{\text{deterministic}} + \underbrace{(R - Q^q(s, v))}_{\text{continuation noise, mean zero}} - \underbrace{(b - \mathbb{E}[b | s])}_{\text{baseline fluctuation, mean zero}},$$

and by Premise P, given (s, v) the last two parts are also independent of the extra random numbers ξ . So the cross terms vanish, and

$$\mathbb{E}[\|(R - b)f_\xi\|^2 | s] = \sum_v q(v | s) \left[(A^q(s, v) + \bar{A}_q(s))^2 + \sigma^2(s, v) + \text{Var}(b | s) \right] \mathbb{E}_\xi \|f_\xi(s, v)\|^2 \quad (23)$$

where $\sigma^2(s, v) = \text{Var}(R | s, v)$ is the variance of the return that comes from the continuation. The variance equals this second moment minus the squared mean $\|G_s[\mu] + \bar{A}_q \bar{f}\|^2$. The last factor is the average size of the multiplier, squared first and then averaged:

$$\mathbb{E}_\xi \|f_\xi\|^2 = \|f_0\|^2 + \text{tr Cov}_\xi(f_\xi),$$

where the second term is the randomness of the multiplier itself. Two knobs act here. When $c = 0$ and w is deterministic, $\|f_0\|^2 = w^2 \|g\|^2$, so tokens with large ratios are amplified. When c is estimated with K independent alternative tokens, $\text{tr Cov}_\xi(f_\xi) = \text{tr } \Sigma_g / K$, and multiplying it by the second moment of the signal gives the V_K of §9.4.

Chapter 9 derives N^* , b_{var} , the effective sample size, and V_K from this one formula. The variance across tokens has cross terms (the tokens of one response share one R); this note analyzes the variance only at a single prefix.

In one sentence: The bias is the sum of three terms, one for each change, and at one prefix the variance is the sum of three kinds of noise times the size of the multiplier.

Next question: First comes A: it is the only term that depends on the baseline, and the only one that the baseline can make worse.

Part I summary: a map of the note and a table of methods

Every term of the umbrella formula is now defined: A, B, and C in Eqs. (19)–(21), D in §3.3, and the variance at one prefix in Eq. (23). Part II (Chapters 4–8) covers the four bias terms one at a time, and Part III (Chapters 9–13) covers the variance, how to choose, a TRPO-style bound, and practice.

Figure 1 can serve as an index for Parts II and III: ellipses are error terms, boxes are methods, each arrow shows whether a method removes, shrinks, or adds a term, and every node names the chapter where it is treated. Table 1 lists what each method does to each term; each cell is derived in a later chapter, and the end of each of Chapters 4–9 says which cells that chapter fills in. The variance column compares with plain REINFORCE on the same prefix, and every entry in it holds only under the conditions stated in the corresponding chapter.

Part II The three bias terms, plus one beyond the premise

4 Term A: what drift is

$$\mathbb{E}\|\hat{g}_N - \nabla_{\theta} J\|^2 = \left\| \mathbf{A} + \mathbf{B} + \mathbf{C} [+ \mathbf{D}] \right\|^2 + \frac{\text{tr Cov}(\hat{g})}{N}$$

This chapter answers: What are the two factors of term A, and why can a group baseline not make the first one zero? Why can A be large when the mismatch is small?

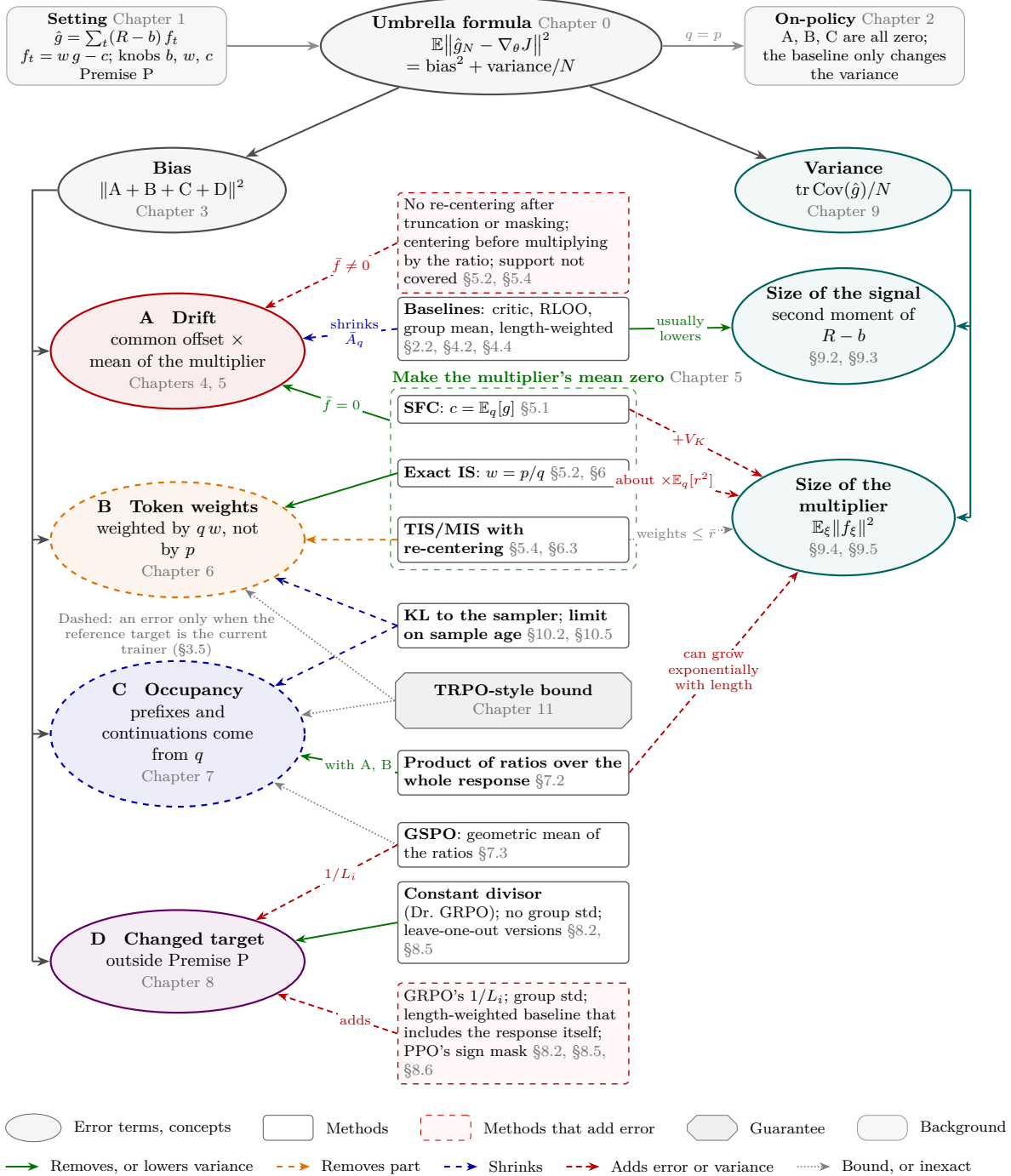
4.1 A number times a direction

Term A, $\mathbf{A} = \sum_t \mathbb{E}_{s_t \sim q} [\bar{A}_q(s_t) \bar{f}(s_t)]$, multiplies at each prefix a **number**, the common offset set by the baseline, by a **direction**, the mean of the multiplier set by (w, c) . For plain REINFORCE the multiplier is g , so by Eq. (16),

$$\text{drift}(s) = \bar{A}_q(s) (m_q(s) - m_p(s)) = -\bar{A}_q(s) \nabla_{\theta} D_{\text{KL}}(q(\cdot | s) \| p_{\theta}(\cdot | s)). \quad (24)$$

So besides the return objective, plain REINFORCE does gradient ascent on $-\bar{A}_q D_{\text{KL}}(q \| p_{\theta})$. The sum $\sum_v q(v) \nabla_{\theta} \log p_{\theta}(v)$ is also the maximum-likelihood gradient on the sampler’s tokens, so drift is $\bar{A}_q$ times one SFT step on the sampler’s outputs, which Marek and Ryabinin [1] call “*distillation toward the sampler*”.

When $\bar{A}_q > 0$ (the baseline is below the value of the prefix, for example at a prefix that is going well, as in Example 1), drift moves the trainer toward the sampler; when $\bar{A}_q < 0$ (the baseline is too high), it moves the trainer away from the sampler. In the local model of Appendix B.2, where



Other chapters: Chapter 10, how to choose; Chapter 12, practice and diagnostics; Chapter 13, questions and answers. Appendices: A, derivations; B, optional mechanisms; C, implementation variants; D, notation.

Figure 1: A map of the note. The umbrella formula splits into bias and variance. The bias splits into four terms, A, B, C, and D (B and C count as errors only when the reference target is the current trainer), and the variance splits into the size of the signal and the size of the multiplier. The middle column lists methods: arrows to the left point to the bias terms a method affects, and arrows to the right show its effect on the variance.

Method	A drift	B weights	C occupancy	D outside Premise P	Variance
Critic as the baseline (§4)	zero if the critic is accurate	unchanged	unchanged	none	often lower, not guaranteed
RLOO; GRPO group mean (§2, §4)	prompt-level part is zero; prefix-level part remains	unchanged	unchanged	none; GRPO only rescales	often lower
Length-weighted baseline (§4, §8)	removes the group-level part on a batch (the “net push”, §4.4)	unchanged	unchanged	self-inclusion length bias (about $1/n$ of the main term for large groups with small shares)	close to the lowest under some assumptions
SFC (§5)	zero for every b	stays at q	unchanged	none	can go up or down; plus V_K when the center is estimated
Exact IS (§5, §6)	zero (needs support coverage)	corrected to p	unchanged	none	usually higher; $\mathbb{E}_q[r^2]$ is a proxy
TIS/MIS without re-centering (§5)	usually smaller, not zero	partly corrected	unchanged	none	weights at most the cap $\bar{r}$
TIS/MIS with re-centering (§5, §6)	zero	partly corrected	unchanged	none	weights at most the cap $\bar{r}$
Whole-response ratio (§7)	zero	zero	zero	none	grows exponentially with the accumulated mismatch
GSPO (§7)	not addressed	not exact	not exact	gradient carries $1/L_i$	weights near 1; the paper reports more stable training
GRPO’s $1/L_i$; group standard deviation (§8)	unchanged	unchanged	unchanged	changes the target	—
PPO’s sign mask (§8)	centering cannot restore baseline invariance	—	—	outside Premise P	—
KL to the sampler; limits on sample age (§10)	smaller	smaller	smaller	none	none

Table 1: What each method does to each term of the umbrella formula. The reference target is the current trainer; if the reference target is the deployed sampler instead, the B and C columns are not errors at all (§1.3, §3.5).

$m_q - m_p$ is roughly proportional to the parameter difference, this second direction can reinforce itself; in general there is no such monotone relation.

The extra objective may help or hurt the return; the problem is that the baseline sets it. Adding a constant Δ to every signal (reward $R + \Delta$, baseline unchanged) does not change the true gradient but adds $\Delta \sum_t \mathbb{E}_q[\tilde{f}(s_t)]$ to the expected update of plain REINFORCE; in Example 1 this is all of the dependence of $-0.0403 + 0.05b$ on b . So the reward encoding matters: with $\{-1, 1\}$ (that is, $2R - 1$) instead of $\{0, 1\}$, the factor 2 only doubles the update, but subtracting 1 adds $-\sum_t \mathbb{E}_q[\tilde{f}]$, unless a baseline such as the group mean shifts with the rewards and cancels it.

4.2 The two parts of the common offset: why we cannot simply choose the right baseline

By Eq. (13), $\bar{A}_q(s_t) = \underbrace{V^q(s_t) - V^q(s_0)}_{\text{(iii)}} + \underbrace{V^q(s_0) - \mathbb{E}[b \mid s_t]}_{\text{(iv)}}$. **A prompt-level baseline removes**

(iv): the leave-one-out average has mean $V^q(s_0)$, as does GRPO’s group mean up to a factor $1 - 1/n$ (Eq. (12)). **It cannot remove (iii)**, which varies with the prefix, while a group has one baseline.

The common offset in the middle of a response

A problem has success rate one half, so $V^q(s_0) = 0.5$ and the group-mean baseline is about 0.5:

Prefix	Success rate from here	$\bar{A}_q$
The key equation is already written correctly	90%	$0.9 - 0.5 = +0.4$
An earlier step was wrong	10%	-0.4

The offset can average to zero over the group, but at these two prefixes it is ± 0.4 . Such offsets are most common halfway through a response: once the outcome is mostly decided, every later token has one.

The baseline may estimate the prompt’s value exactly, but it is used at intermediate prefixes whose value has changed. So (iii) is not sampling noise, and it does not go away however large the group is; only a critic that values each prefix can make it smaller, which for long responses is hard and expensive. **So in RL for language models the common offset basically cannot be removed; what we can do is make the mean of the multiplier zero** (Chapter 5).

GRPO as used in practice [identity]. GRPO’s group mean includes this response’s own reward, and GRPO also divides by the group standard deviation, so (P1) does not hold. Because the other responses are independent of this one (Appendix A.11),

$$\mathbb{E}[A_i \mid s, v] = \mathbb{E}_q[\chi(R_i) \mid s, v], \quad \chi(R_i) = \mathbb{E}[A_i \mid R_i],$$

where $\chi(R)$ is the signal averaged over the other responses with this response’s reward fixed. So the derivation holds with $R - b$ replaced by $\chi(R)$ and the baseline set to zero. The signals of a group add up to zero and the responses play the same role, so $\mathbb{E}[\chi(R)] = 0$: (iv) is zero, but (iii) remains at intermediate prefixes. The form of χ depends on the normalization:

- group mean only: $\chi(R) = (1 - 1/n)(R - V^q(s_0))$;
- also dividing by the standard deviation, two-valued reward: $\chi(R) = \zeta (R - V^q(s_0))$, where $\zeta > 0$ depends only on n and $V^q(s_0)$ (for $n = 8$, about 1.86 at $V^q(s_0) = 0.5$ and 2.11 at 0.2 or 0.8); within a prompt this is a fixed baseline times a constant, but across prompts it weights extreme success rates more: the difficulty bias of Dr. GRPO [12] (§8.5);
- three or more reward values: χ can be nonlinear, so the update learns the advantage of $\chi(R)$, not of R ; this is term D (Chapter 8).

4.3 The mismatch is small, but A can be large

The distance between q and p alone understates A; what matters is the ratio of A to the signal term, which comes from the real differences between tokens:

$$\frac{\|A\|}{\|\text{signal term}\|} \approx \frac{|\bar{A}_q| \cdot \|m_q - m_p\|}{\|\text{Cov}_q(A^q, \nabla_\theta z)\|}.$$

Here $|\bar{A}_q|$ can be of order 0.4, while A^q is often of order 0.01 at a good prefix. In Example 1, $D_{\text{KL}}(q\|p)$ is only 0.0054, but the tokens differ by only 0.02 while the common offset is 0.4, so A is 4.2 times the signal term and the update points the wrong way.

In one sentence: A small KL does not make A small relative to the signal: when the tokens differ little and the common offset is large, a tiny mismatch can give the update the wrong sign.

4.4 The drift of a group of responses: two levels

Now take a batch with a token-level loss (a direct sum over all tokens). Splitting each token’s score as $g = h + (m_q - m_p)$, where $h = \nabla_\theta z - m_q$ is the SFC score of the next chapter and has mean zero under q , the drift part of the update is $A_i u_{it}$, where $u_{it} = \bar{f}(s_{it}) = (m_q - m_p)(s_{it})$ is the mean of the multiplier at the prefix of token t of response i . With $\bar{u}$ its average over the group,

$$\sum_{i,t} A_i u_{it} = \underbrace{\bar{u} \sum_{i,t} A_i}_{\text{group level: common direction times net push}} + \underbrace{\sum_{i,t} A_i (u_{it} - \bar{u})}_{\text{prefix level}}. \quad (25)$$

We call $\sum_{i,t} A_i = \sum_i L_i A_i$ the **net push**. With the group mean $b = \bar{R}$,

$$\sum_i L_i (R_i - \bar{R}) = \sum_i (L_i - \bar{L})(R_i - \bar{R}) = n \text{Cov}_G(L, R), \quad (26)$$

where $\bar{L}$ is the mean length and Cov_G the within-group covariance of length and reward (dividing by n). It is often negative in reasoning and agent tasks, for example when truncated responses get zero reward or a length penalty makes long responses worse, and the net push then has the sign that moves the trainer away from the sampler (Appendix B.4).

What a group-level scalar baseline can do. For a batch that has already been sampled, with

the u_{it} held fixed, the drift of the group with baseline b is

$$\mathcal{D}(b) = \sum_{i,t} (R_i - b) u_{it} = \underbrace{\sum_{i,t} R_i u_{it}}_{\mathcal{D}_R} - b \underbrace{\sum_{i,t} u_{it}}_U. \quad (27)$$

This is linear in b , so a scalar baseline can move it only along U ; when $U \neq 0$, $\|\mathcal{D}(b)\|$ is smallest at $b^* = \langle \mathcal{D}_R, U \rangle / \|U\|^2$. If all the u_{it} point in the same direction, write $U_i = \sum_t u_{it}$ for the total drift of the i -th response; then

$$b^* = \frac{\sum_i R_i U_i}{\sum_i U_i}, \quad \mathcal{D}(b^*) = 0,$$

the mean reward weighted by total drift. If moreover every token has the same drift, $U_i \propto L_i$ and b^* is the **length-weighted mean** $b_{\text{LW}} = \sum_i L_i R_i / \sum_i L_i$, which makes the net push $\sum_i L_i (R_i - b_{\text{LW}})$, the first term of Eq. (25), exactly zero.

One long failure and one short success

This case is common when there is a length limit and truncated responses get a reward of 0.

	Length L_i	Reward	A_i , group mean ($b = 0.5$)	A_i , length-weighted ($b = 1/6$)
Response 1	100	0	-0.5	-1/6
Response 2	20	1	+0.5	+5/6
$\sum_i A_i$			0	2/3
$\sum_i L_i A_i$			-50 + 10 = -40	-16.7 + 16.7 = 0

With the group mean and the same u_{it} at every token, the drift is -40 times this vector, and the signal per token averages -1/3, exactly $\text{Cov}_G(L, R) / \bar{L} = -20/60$; length weighting makes it zero.

Length weighting does not guarantee a smaller drift. It uses token counts as a proxy for each response’s drift, so if the drift varies a lot across tokens it can do worse than the group mean.

Counterexample: drift is not spread evenly over the tokens

Most tokens of the long response are nearly deterministic ($p \approx q \approx 1$) and have little drift, while at every token of the short one the sampler and the trainer differ a lot. Along one direction:

	Length L_i	Reward	Drift per token	Total drift U_i
Response 1	100	0	0.001	0.1
Response 2	20	1	0.05	1.0

So $\mathcal{D}(b) = (0 - b)(0.1) + (1 - b)(1.0) = 1 - 1.1b$: 0.45 for the group mean ($b = 0.5$), a larger 0.82 for length weighting ($b = 1/6$), and 0 for weighting by total drift ($b^* = 1/1.1 \approx 0.91$).

No scalar baseline can change the prefix-level part. Even with a zero net push, the second term of Eq. (25), $\sum A_i (u_{it} - \bar{u})$, is in general not zero when the u_{it} differ across prefixes.

Number times direction: a zero net push is not enough

Three tokens, with the logits as the parameters. One group has:

Tokens from	Number	Advantage	u at their prefixes
Responses that end up correct	100	+0.6	(0.1, 0, -0.1)
Responses that are wrong	150	-0.4	(0.04, 0, -0.04)

The net push is $100 \times 0.6 - 150 \times 0.4 = 0$, as the length-weighted baseline guarantees. If u were (0.1, 0, -0.1) at every prefix, the drift would be $0 \times (0.1, 0, -0.1) = 0$; with the u of the table it is $60(0.1, 0, -0.1) - 60(0.04, 0, -0.04) = (3.6, 0, -3.6)$: A sums numbers times directions.

So a group-level baseline acts on the number factor, and it removes the drift of a group only when the drift has the same direction everywhere and is in proportion to the number of tokens; removing drift at every prefix needs the direction factor to be zero. Also, a b computed from the same batch depends on these tokens, so (P1) no longer holds strictly.

How the three loss conventions behave here.

- The original GRPO [10] averages each response’s tokens with $1/L_i$, so each response has total weight one and the group mean makes $\sum_i A_i = 0$: zero net push, at the cost of the length bias of $1/L_i$ itself (Chapter 8).
- DAPO’s token-level loss [11] and Dr. GRPO’s constant divisor remove $1/L_i$, and with it the zero net push, which becomes $n \text{Cov}_G(L, R)$.
- The length-weighted baseline restores it, at the cost of a self-inclusion length bias (§8.3).

With SFC, the net push does not affect the bias.

4.5 Drift is systematic

A is part of the expected update, so more samples from the same distribution do not average it out. Its direction varies, because $\bar{A}_q$, q , p , and the logit Jacobian change across prefixes and steps; but where it is correlated across many positions or steps, it can accumulate. Appendix B gives the conditions and the mechanisms involved (response length, the staleness feedback loop, entropy, the within-group correlation of length and reward), under stated model assumptions.

Cells of Table 1 filled in this chapter: Column A: critic, RLOO/GRPO, length-weighted.

What remains after this term: The number factor cannot be made zero at every prefix, so the next chapter makes the direction factor zero: SFC, exact IS, and truncation with re-centering.

5 Removing term A: making the mean of the multiplier zero

$$\mathbb{E} \|\hat{g}_N - \nabla_{\theta} J\|^2 = \left\| \mathbf{A} + \mathbf{B} + \mathbf{C} [+ \mathbf{D}] \right\|^2 + \frac{\text{tr Cov}(\hat{g})}{N}$$

This chapter answers: How do SFC, exact IS, and truncation with re-centering make the mean of the multiplier zero, and which practices make it nonzero again?

By Eq. (19), if the mean of the multiplier under the sampler, $\bar{f}(s) = \mathbb{E}_q[wg \mid s] - c(s)$, is zero, A is zero however far off the baseline is. Either set $c = 0$ and choose w so that $\mathbb{E}_q[wg] = 0$ (exact IS), or choose any w and take $c = \mathbb{E}_q[wg]$ (SFC with $w = 1$, truncation with re-centering with $w = \tilde{w}$).

5.1 SFC: change the centering

SFC subtracts the score’s average under the sampler:

$$h(s, v) = g(s, v) - \mathbb{E}_q[g \mid s] = \nabla_\theta z(s, v) - m_q(s), \quad \mathbb{E}_q[h \mid s] = 0. \quad (28)$$

The second equality uses $g = \nabla_\theta z - m_p$ and $\mathbb{E}_q[g] = m_q - m_p$. This is the fix of Marek and Ryabinin [1], who call it “*score centering*”; Zhang’s blog series [3] calls it **score-function centering (SFC)**. It centers with the sampler’s average m_q instead of the trainer’s m_p , so sampling and centering use the same distribution; in the notation of §1.4, $w = 1$, $c = m_q - m_p$, $\mu = q$.

In Example 1, $h(1) = 1 - 0.6 = 0.4$ and $h(2) = 0 - 0.6 = -0.6$, so the expected update is $0.6(0.91 - b)(0.4) + 0.4(0.89 - b)(-0.6) = 0.0048$ for every b : the part of the signal the two tokens share (about 0.4 at $b = 0.5$) cancels, without the algorithm needing to know that the prefix’s success rate is 0.902.

The implementation does not need q over the whole vocabulary. At the same prefix the sampler draws K extra tokens $a'_1, \dots, a'_K$ (“*alternative tokens*”), unrelated to the main token’s outcome; the loss is

$$\mathcal{L}_{\text{SFC}} = -\text{sg}(A) \left[\log p(a) - \frac{1}{K} \sum_{k=1}^K \log p(a'_k) \right] = -\text{sg}(A) \left[z(a) - \frac{1}{K} \sum_{k=1}^K z(a'_k) \right], \quad (29)$$

where sg is stop-gradient (the signal and the sampled tokens are held fixed). The second form holds because $\log p(v) = z(v) - \log Z_p$, and the trainer’s log-normalizer $\log Z_p$ has coefficient 1 on the main token and $K/K = 1$ on the subtracted part. The negative gradient, $-\nabla_\theta \mathcal{L}_{\text{SFC}} = A(\nabla_\theta z(a) - \frac{1}{K} \sum_k \nabla_\theta z(a'_k))$, has expectation $Ah(a)$ over the alternative tokens, which must have the same distribution as the main token (same temperature and truncation), be drawn with replacement, and be independent of its outcome (§9.4).

Three ways to get the center. The logit form still needs the sampler, because the center $m_q = \mathbb{E}_q[\nabla_\theta z]$ is an average under it. There are three ways to get it, with errors of different kinds:

How the center is obtained	Relative to the true sampler	Main error
Compute $\sum_v q(v) \nabla_\theta z(v)$ from the sampler’s full q	Exact	None from the center; needs the full q
Average K alternative tokens drawn independently from q (Eq. (29))	Unbiased estimate	Extra variance V_K (§9.4)
Logged head of q , with the tail split as the trainer splits it	Exact centering for an approximate $\hat{q}$	Can leave a fixed bias

For the third way, let H be the logged head (for example, the sampler’s most likely tokens), $\bar{H}$ the tail, and $q(\bar{H})$, $p(\bar{H})$ the tail’s total probability under the sampler and the trainer. If $p(\bar{H}) > 0$, it uses

$$\hat{q}(v) = \begin{cases} q(v), & v \in H, \\ q(\bar{H}) \frac{p(v)}{p(\bar{H})}, & v \in \bar{H}. \end{cases} \quad (30)$$

So the tail’s total probability comes from the sampler, but the trainer’s conditional distribution $p(\cdot | \bar{H})$ splits it; this is neither the trainer’s tail as it is nor an unbiased sample of the sampler’s tail. Centering with $\hat{q}$ leaves this mean of the multiplier under the true q (with both tail probabilities positive; $q(\cdot | \bar{H})$ and $p(\cdot | \bar{H})$ are the two distributions restricted to the tail and renormalized):

$$\mathbb{E}_q[g - \mathbb{E}_{\hat{q}}[g]] = \sum_{v \in \bar{H}} \left(q(v) - q(\bar{H}) \frac{p(v)}{p(\bar{H})} \right) g(v) = q(\bar{H}) \left(\mathbb{E}_{q(\cdot | \bar{H})}[g] - \mathbb{E}_{p(\cdot | \bar{H})}[g] \right). \quad (31)$$

The first equality holds because $\hat{q} = q$ on the head. The residual is small if the tail probability is small and the tail tokens’ scores are bounded, or if the two tail distributions have similar mean scores; a small tail probability alone is not enough. It is zero if $q(\bar{H}) = 0$, that is, if the head covers every token the sampler can draw.

Marek and Ryabinin [1] use this approximation, and their code logs the $k = 128$ most likely tokens by default [2]; Appendix A.4 of their paper reports that $k = 128$, and even $k = 32$, works as well as the full q in every setting they tried. The full loss and its weighted version are in Appendix C.1.

Why we subtract this average: probability must be conserved

The score of a distribution averages to zero under it: $\sum_v q(v) \nabla_{\theta} \log q(v) = \sum_v \nabla_{\theta} q(v) = \nabla_{\theta} \sum_v q(v) = 0$. The trainer’s score averaged under q , $\sum_v q(v) g(v) = m_q - m_p$, is in general not zero: the update raises or lowers the log-probabilities of the sampler’s tokens together, moving the trainer as a whole toward or away from the sampler. Subtracting $\mathbb{E}_q[g]$ removes this shared rise or fall, leaving only shifts of probability between tokens. And h is itself a score: that of the sampler tilted by the trainer’s relative change, $\tilde{q}_{\delta}(v) \propto q(v) p_{\theta+\delta}(v)/p_{\theta}(v)$, at $\delta = 0$ (Appendix A.6).

5.2 Exact IS: change the weight

Exact IS weights the trainer’s score by the ratio $r(v) = p(v)/q(v)$, so $f = r g$. Multiplying by r turns an average under q into an average under p , where the trainer’s score averages to zero:

$$\mathbb{E}_q[r g] = \sum_v q(v) \frac{p(v)}{q(v)} g(v) = \mathbb{E}_p[g] = 0.$$

In Example 1, $r(1) = 0.65/0.6 \approx 1.083$ and $r(2) = 0.35/0.4 = 0.875$, so the expected update is $0.6 \times 1.083 \times (0.91 - b)(0.35) + 0.4 \times 0.875 \times (0.89 - b)(-0.65) = 0.00455$, also for every b .

The support condition. The second equality requires $q(v) > 0$ wherever $p(v) > 0$. If the sampler cannot draw some tokens that the trainer would choose, then $\mathbb{E}_q[r g] = -\sum_{v: q(v)=0} p(v) g(v)$, in general not zero, and A comes back. Three things need to be kept apart:

Practice or condition	What it solves	What it does not solve by itself
Compute q with the actual temperature and filtering rules	The wrong distribution is not used as q	That q covers the trainer's support
$\text{supp}(p) \subseteq \text{supp}(q)$, with the correct ratio and the trainer's score	Exact IS is unbiased for the specified target	The variance can be large (Chapter 9)
Re-center the weighted score	Restores baseline invariance	Tokens with $q = 0$ and $p > 0$

Two points are often misread. With a temperature other than 1 and no truncation, the softmax is positive on every token, so the support is fine; only q must be the true, temperature-scaled distribution. The same top- k or top- p settings do not give the same support, because the two sides' logits differ and they can keep different tokens. The sampler's actual mask can define a restricted target (the trainer renormalized on the kept set) for exact IS, but then the target has changed, which must be stated, and the restricted distribution's own score must be used.

SFC does not need this condition, because $\mathbb{E}_q[h] = 0$ uses only q ; tokens with $q = 0$ get zero weight in the SFC update, which belongs to term B.

5.3 The two methods end up with different weights

SFC and exact IS both make A zero, yet in Example 1 they give 0.0048 and 0.00455, because the weights differ: SFC has $\mu = q$ and exact IS has $\mu = p$. The difference is term B (Chapter 6).

5.4 Two practices that make the mean of the multiplier nonzero again

Truncating or masking the ratio without re-centering. In practice IS often truncates the ratio, $\tilde{w} = \min(r, \bar{r})$ with a ratio cap $\bar{r} \geq 1$ (truncated IS, TIS [7]), or sets the weight to zero for tokens whose ratio is outside an interval $[r_{\min}, r_{\max}]$ (masked IS, MIS; IcePop [8] at the token level, [9] at the sequence level). This lowers the weights of the tokens the trainer favors most, so the mean is no longer zero (Appendix A.9):

$$\mathbb{E}_q[\tilde{w}g] = - \sum_{v: r(v) > \bar{r}} (p(v) - \bar{r}q(v))g(v) \neq 0. \quad (32)$$

In Example 1 both ratios are below 2, so truncation does nothing. In Example 2 of Chapter 6 ($q = (0.1, 0.1, 0.8)$, $p = (0.8, 0.1, 0.1)$, the logits as parameters), the ratios are $(8, 1, 0.125)$; with $\bar{r} = 2$, $\tilde{w} = (2, 1, 0.125)$ and $\mathbb{E}_q[\tilde{w}g] = (-0.12, 0.06, 0.06)$, against $\mathbb{E}_q[g] = q - p = (-0.7, 0, 0.7)$ without weighting: shorter, but not zero. Truncation usually makes A smaller, but not always (Appendix A.9 has a counterexample). The fix is to **weight first, then center**: $f = \tilde{w}g - \mathbb{E}_q[\tilde{w}g]$ has mean zero again.

Reversing the order. Centering with SFC first and then multiplying by the ratio, $f = r h$, moves the average to p while h was centered under q : $\mathbb{E}_q[r h] = \mathbb{E}_p[h] = m_p - m_q \neq 0$, so A comes back with the opposite sign. This multiplier is not in the class of §1.4 (it is not wg minus a vector that depends only on the prefix), so we compute its expectation directly:

$$\mathbb{E}_q[(R - b) r h] = G_s[p] + (\mathbb{E}_p[Q^q] - \mathbb{E}[b]) (m_p - m_q).$$

The coefficient is $\mathbb{E}_p[Q^q] - \mathbb{E}[b]$ rather than $V^q - \mathbb{E}[b]$; the two differ by $\mathbb{E}_p[A^q]$, the trainer’s one-step improvement over the sampler. In Example 1 this gives $0.0497 - 0.05b$. With an exact ratio, the correct order subtracts $\mathbb{E}_q[r g] = 0$, so “exact IS plus SFC” done right is just exact IS.

After weighting, use log-probabilities, not logits. In Eq. (29), $\log Z_p$ cancels sample by sample. With a token-dependent weight, its coefficient is $\tilde{w}(a)$ on the main token but the average weight on the subtracted part, equal only on average, with a difference correlated with the signal. So the logit form adds $\text{Cov}_q(A, \tilde{w}) m_p$ to the expected update (Appendix A.10). This term does not change when the baseline is shifted, so it is not A, but it changes the update: in Example 2 it is $\mathbb{E}_p[A^q] m_p = (0.56, 0.07, 0.07)$ with the exact ratio and $0.19 m_p = (0.152, 0.019, 0.019)$ with truncation at $\bar{r} = 2$, larger in both cases than the signal term (with truncation, $G[\min(p, 2q)]$ has length 0.042).

5.5 The multipliers at a glance [identity]

Method	Multiplier f	Mean $\bar{f}$	Term A	In Example 1
Plain REINFORCE	g	$m_q - m_p$	Yes	$-0.0403 + 0.05b$
SFC	$h = g - \mathbb{E}_q[g]$	0	No	0.0048
Exact IS	$r g$	$\mathbb{E}_p[g] = 0$	No	0.00455
Ratio times SFC score (wrong order)	$r h$	$m_p - m_q$	Yes, opposite sign	$0.0497 - 0.05b$
Truncated IS (TIS)	$\tilde{w} g$	$-\sum_{r > \bar{r}} (p - \bar{r}q) g$	Yes, usually smaller	Example 2 (Chapter 6)
TIS with re-centering	$\tilde{w} g - \mathbb{E}_q[\tilde{w} g]$	0	No	Same as above

The exact IS row assumes support coverage. The first four rows form a 2×2 table, the distribution the samples are averaged under (q , or p through the ratio) $\times$ the distribution the score is centered under (q or p): the diagonal cells (SFC, exact IS) have no term A, and the off-diagonal cells do.

5.6 Number times direction: how the baseline and SFC work together

Of the two factors of A, the length-weighted baseline changes the number (zero net push) and SFC the direction (zero mean at every prefix). Combined, four facts hold:

1. **Without SFC, length weighting is only a partial fix.** It makes the group-level part of Eq. (25) zero, but the prefix-level part remains.
2. **With SFC, length weighting is not needed for the bias, but it still reduces the variance.** A is zero at both levels. The remaining SFC noise $\sum_{i,t} A_{it} e_{it}$ (A_{it} is the signal of token t of response i , equal to A_i when the reward is not assigned per token; e_{it} is the error of the estimated center at this prefix) has variance roughly proportional to $\sum_{i,t} A_{it}^2$, and the constant b minimizing $\sum_{i,t} (R_i - b)^2$ is the per-token least-squares average b_{LW} (Appendix A.12). So among constant baselines it gives the least SFC noise, and the group mean more, if the multiplier has about the same size at every token.

3. **SFC does not remove the self-inclusion bias of length weighting.** That bias exists already on-policy, in the signal term: b_{LW} depends on this response's own reward and length, so $\text{Cov}_q(R - b_{\text{LW}}, f)$ differs from $\text{Cov}_q(R, f)$. With a large group, small shares, and bounded moments of the reward and the score, it is $O(1/n)$ relative to the main term, but this says little in directions where the main gradient is close to zero (§8.2). Removing it takes another baseline: the leave-one-out length-weighted one, or the plain group mean (its self-inclusion is the same for every response, so it only scales the update by $1 - 1/n$, but it gives up the zero net push).
4. **The leave-one-out length-weighted baseline does not remove the off-policy A.** It changes the number, not the direction, and gives up the exact zero net push: what remains is about $1/n$ of the group mean's net push in a large group with small shares, and can be much more in small groups or when one response has a large share, as in the four-response example of Chapter 8 ($n = 4$, one response has half of the tokens): -2413 , even larger in magnitude than the -2000 of the group mean. With SFC this is only variance, and SFC noise stays near the lowest.

Cells of Table 1 filled in this chapter: Column A of the SFC, exact IS, and TIS/MIS rows.

What remains after this term: The expected update is now $\sum_t \mathbb{E}_q[G[\mu]]$, the advantage update with weights μ ; the next chapter asks how much it matters that SFC has $\mu = q$ and exact IS $\mu = p$.

6 Term B: the token weights

$$\mathbb{E} \|\hat{g}_N - \nabla_{\theta} J\|^2 = \left\| \mathbf{A} + \mathbf{B} + \mathbf{C} + \mathbf{D} \right\|^2 + \frac{\text{tr Cov}(\hat{g})}{N}$$

This chapter answers: Why do two methods without term A give different updates, and how large is the gap? How much of it does a truncated ratio correct?

6.1 What the weight changes: whose average a token is compared with

Once the mean of the multiplier is zero, the expected update at a prefix is only the signal term $G_s[\mu] = \sum_v \mu(v) A^q(v) g(v)$. For the three methods without term A:

- **SFC**, $f = h$: $\mathbb{E}_q[A^q h] = \mathbb{E}_q[A^q g] - \mathbb{E}_q[A^q] \mathbb{E}_q[g] = G_s[q]$, because $\mathbb{E}_q[A^q] = 0$.
- **Exact IS**, $f = r g$: $\mathbb{E}_q[A^q r g] = \sum_v p(v) A^q(v) g(v) = G_s[p]$ (assuming support coverage).
- **TIS with re-centering**, $f = \tilde{w} g - \mathbb{E}_q[\tilde{w} g]$: $\mathbb{E}_q[A^q f] = \mathbb{E}_q[A^q \tilde{w} g] = G_s[\min(p, \bar{r} q)]$, because $q \cdot \min(p/q, \bar{r}) = \min(p, \bar{r} q)$.

In terms of covariances, $G_s[q] = \text{Cov}_q(A, \nabla_{\theta} z)$ and $G_s[p] = \text{Cov}_p(A, \nabla_{\theta} z)$, and the baseline cancels in both, so B is just the difference between two covariances. With the trainer as the reference target, we compare with its own on-policy update with the same signal, $\mathbb{E}_p[Ag] = G_s[p]$; for a

method with weights μ , the gap at this prefix is

$$G_s[\mu] - G_s[p] = \sum_v (\mu(v) - p(v)) A^q(v) g(v), \quad (33)$$

and summing it over prefixes gives B. The gap of SFC is $\sum_v (q - p) A^q g$; exact IS has no gap. But when this inference engine is what is deployed, the fixed-error model holds, and the centering is accurate (§3.5), the reference target is the sampler’s gradient $\nabla_\theta J(q_\theta) = \sum_t \mathbb{E}_q[G[q]]$, which SFC gives exactly; so this gap is an error only when the reference target is the trainer.

The intuitive difference between the two weights. In the tabular model (the parameters are the logits, $\nabla_\theta z(v) = e_v$, $g(v) = e_v - p$), the u -th component of $G_s[\mu]$ is

$$G_s[\mu]_u = \mu(u) A^q(u) - p(u) \sum_v \mu(v) A^q(v).$$

When $\mu = q$, the sum is $\mathbb{E}_q[A^q] = 0$, so $G_s[q]_u = q(u)(Q^q(u) - V^q)$: token u is compared with **the sampler’s average** and weighted by the sampler’s probability. When $\mu = p$, $G_s[p]_u = p(u)(Q^q(u) - \sum_v p(v) Q^q(v))$: token u is compared with **the trainer’s average** and weighted by the trainer’s probability. In Example 1 (two actions, $\nabla_\theta z \in \{1, 0\}$) the covariance reduces to $\pi(1)\pi(2)(Q(1) - Q(2))$: SFC gets $0.6 \times 0.4 \times 0.02 = 0.0048$ and exact IS gets $0.65 \times 0.35 \times 0.02 = 0.00455$.

Example 2: three tokens, a very stale sampler

The sampler rarely picks the best token, and the trainer has already learned to pick it. The sampler’s average value is $V^q = 0.15$, and the trainer’s is $\sum_v p(v) Q^q(v) = 0.85$. The parameters are the three logits, and the KL between the two distributions is 1.46, a large lag. The last two columns are the updates to the logits:

Token	q	p	Q^q	A^q	SFC, $G[q]$	Exact IS, $G[p]$
1 (best)	0.1	0.8	1	0.85	+0.085	+0.120
2 (middle)	0.1	0.1	0.5	0.35	+0.035	−0.035
3 (worst)	0.8	0.1	0	−0.15	−0.120	−0.085

Token 2 is better than the sampler’s average ($0.5 > 0.15$) but worse than the trainer’s ($0.5 < 0.85$), so SFC raises its probability and exact IS lowers it. Both updates have length 0.151, and the distance between them is 0.086.

6.2 How large the gap is

When q and p are close, the gap is small. With $B_{\max} = \max_v \|A^q(v)g(v)\|$ and $A_{\max} = \max_v |A^q(v)|$,

$$\begin{aligned} \|G_s[p] - G_s[q]\| &\leq B_{\max} \sqrt{2 D_{\text{KL}}(q||p)} \quad (\text{gradient, Appendix A.8}), \\ \text{gap in the objective} &\leq A_{\max} D_{\text{KL}}(q||p) \quad (\text{Appendix A.7}). \end{aligned}$$

These two inequalities always hold. Reading them as “first order” and “second order” also needs three conditions:

- q and p have the same support;
- q is strictly positive on this support;
- p is a small, smooth perturbation of q (for example, logits that differ by a small δ).

Then $D_{\text{KL}}(q\|p)$ is second order in δ , so the gap is first order in δ for the gradient and second order for the objective.

At the boundary of the support this reading fails. With $q = (1, 0)$, $Q^q = (0, 1)$ (so $V^q = 0$, $A^q = (0, 1)$) and a trainer $\pi_\nu = (1 - \nu, \nu)$ that puts probability ν on the token the sampler cannot draw, the linear surrogate $\mathbb{E}_{\pi_\nu}[A^q] = \nu$ and the log surrogate of SFC, $\mathbb{E}_q[A^q \log(\pi_\nu/q)] = 0$, differ by ν , and $D_{\text{KL}}(q\|\pi_\nu) = -\log(1 - \nu) = \nu + O(\nu^2)$ is first order too: the bound holds, but the KL is no longer second order (Appendix A.7).

6.3 Truncation with re-centering: only a partial correction

The weight of TIS with re-centering is $\min(p, \bar{r}q)$. Substituting it into Eq. (33) gives

$$G_s[p] - G_s[\min(p, \bar{r}q)] = \sum_{v: r(v) > \bar{r}} (p(v) - \bar{r}q(v)) A^q(v) g(v). \quad (34)$$

Tokens with $r \leq \bar{r}$ are corrected exactly: all the tokens the trainer has moved away from ($r < 1$) and those it moderately prefers. Tokens with $r > \bar{r}$, which have the largest weights and the largest variance, are corrected only partly: their shortfall shrinks from $p - q$ to $p - \bar{r}q$.

Every shortfall gets smaller, but the overall gap need not, for two reasons. First, the remaining shortfalls are all positive and add up to $1 - \bar{w}$, where $\bar{w} = \sum_v \min(p, \bar{r}q) = \mathbb{E}_q[\tilde{w}] < 1$ is the total mass of the weights; since $G_s[\min(p, \bar{r}q)] = \bar{w} \cdot G_s[\min(p, \bar{r}q)/\bar{w}]$, the update gets shorter, and the shape of its weights changes too. Second, the shortfalls $p - q$ of SFC add up to zero and partly cancel; truncation removes the negative ones, and with them this cancellation.

Example 2 (continued): the weights and the update after truncation at $\bar{r} = 2$

Token	1	2	3
Sampler q	0.1	0.1	0.8
Trainer p	0.8	0.1	0.1
Ratio $r = p/q$	8	1	0.125
Truncated ratio $\tilde{w} = \min(r, 2)$	2	1	0.125
Weight $\mu = q \tilde{w} = \min(p, 2q)$	0.2	0.1	0.1
True advantage A^q	0.85	0.35	-0.15
Update $G[\mu]$	+0.018	+0.016	-0.034

The total mass of the weights is $\bar{w} = 0.4$. By the component formula above, $\sum_v \mu A^q = 0.17 + 0.035 - 0.015 = 0.19$, so the first component is $0.17 - 0.8 \times 0.19 = 0.018$, and so on. The update has length $\sqrt{0.018^2 + 0.016^2 + 0.034^2} = 0.042$, against 0.151 for $G[q]$ and $G[p]$, and its distance from $G[p]$ is 0.125, even larger than the 0.086 of SFC (with $\bar{r} = 4$ it is 0.083). Most of

the gap is on token 1 ($r = 8 > \bar{r}$), exactly the part that truncation gives up.

Example 2 is an unfavorable case. In the random toy models of §9.6 with a moderate lag, truncation with re-centering is closer to $G[p]$ than SFC in 90% or more of the cases, and the median ratio of its gap to that of SFC is about 0.2 ($\kappa = 2$) and 0.5 ($\kappa = 4$).

Cells of Table 1 filled in this chapter: Column B of the SFC, exact IS, and TIS/MIS rows.

What remains after this term: Even with weights p at every prefix (exact IS), the sampler still decides which prefixes appear and how each is continued. This is term C.

7 Term C: occupancy

$$\mathbb{E}\|\hat{g}_N - \nabla_{\theta} J\|^2 = \left\| \text{A} + \text{B} + \text{C} [+ \text{D}] \right\|^2 + \frac{\text{tr Cov}(\hat{g})}{N}$$

This chapter answers: Once the weights at every prefix are corrected, what is still wrong with the update? What is the exact fix, and why is it not used in practice?

7.1 Two sources

So far we have looked at one prefix, but the full gradient adds up many prefixes, each weighted by how often the policy visits it. Rollouts from q differ from the trainer’s own in two more ways, C1 and C2 of §3.3:

- **C1, which prefixes appear:** prefixes appear as often as the sampler visits them, not as often as the trainer would.
- **C2, the continuation after the prefix:** the return, and so the advantage, comes from the sampler finishing the response; the data contain A^q , while the true gradient needs A^p .

SFC and per-token ratios change only how the tokens are weighted **at a given prefix**, so they change neither of these; TIS and MIS, which multiply each token only by the (truncated or masked) ratio of its own step, fix B but not C. With the sampler as the reference target, C does not exist (§3.5).

Example 3: two solution routes (C1)

At step 1 the model chooses a route: X (“set up equations”) or Y (“try small numbers”). The stale sampler picks X with probability 0.9, the current trainer with probability 0.5. At step 2 the model finishes the solution along the chosen route, and the two policies behave exactly the same there, so B is zero and the per-token ratio is 1. But 90% of the step-2 training data come from route X, while the trainer’s own gradient weights the routes 50/50, so “how to finish route Y well” gets one fifth of the weight it should. The fix is to multiply the step-2 tokens by the step-1 ratio, $p(X)/q(X) = 0.56$ or $p(Y)/q(Y) = 5$.

Example 3 (continued): the continuation itself can also be wrong (C2)

Now let both policies pick X and Y with probability one half each, so the prefix distributions are the same and there is no earlier ratio to multiply by. But on route X the sampler’s success rate on the rest of the solution is 20% and the trainer’s is 60%; on route Y both are 50%. With the advantage from the sampler’s continuations, the step-1 update lowers X , $q(X) A^q(X) = 0.5 \times (0.2 - 0.35) = -0.075$, while the trainer’s own gradient raises it, $0.5 \times (0.6 - 0.55) = +0.025$. Only the ratios of the later steps can correct this, so at the level of the gradient C1 and C2 are two separate first-order errors.

7.2 The exact fix: multiplying the ratios over the whole response

Take the weight of every token to be the ratio of the whole response,

$$w^{\text{seq}} = \prod_{t'=1}^L \frac{p(y_{t'} | s_{t'})}{q(y_{t'} | s_{t'})} = \frac{p(y | x)}{q(y | x)},$$

which is the same for all t . It depends on the whole response, so it is not in the class of §1.4; computed directly, it turns an expectation under q into one under p :

$$\mathbb{E}_{y \sim q} \left[\frac{p(y | x)}{q(y | x)} (R - b) g(s_t, y_t) \right] = \mathbb{E}_{y \sim p} [(R - b) g(s_t, y_t)] = \mathbb{E}_{s_t \sim p} \left[\sum_v p(v | s_t) A^p(s_t, v) g(s_t, v) \right], \quad (35)$$

which is exactly the t -th term of the true gradient. So A, B, and C all go away at once, and the position of each ratio shows which term it removes:

- the ratios of the steps before t change the prefix distribution to that of p (C1);
- the ratio of step t itself changes the weights to p (B);
- the ratios of the steps after t change the continuation to that of p , so Q^q becomes Q^p (C2);
- and $\sum_v p(v)g(v) = 0$ makes the contribution of the baseline zero (A).

Appendix A.16 checks this numerically on a two-step toy model.

The cost. The second moment of the whole-response ratio has an exact form:

$$\mathbb{E}_{y \sim q} [(w^{\text{seq}})^2] = \sum_y \frac{p(y | x)^2}{q(y | x)} = \exp(D_2(p \| q)),$$

where D_2 is the Rényi divergence of order 2 over whole responses. Because the ratio at each step depends on what was chosen before, this is in general not a product of the unconditional per-step second moments $\mathbb{E}_q[r_t^2]$. In an explicit model, the length is fixed at L and the ratio has the same conditional second moment at every prefix, $\mathbb{E}_q[r_t^2 | s_t] = 1 + \sigma_r^2$; since its conditional mean is always 1, σ_r is the conditional standard deviation of the per-step ratio (not the return variance $\sigma^2(s, v)$). Taking conditional expectations from the last step backward gives

$$\mathbb{E}_q [(w^{\text{seq}})^2] = (1 + \sigma_r^2)^L,$$

which grows exponentially with the length; the ratio ESS (§9.4, $N/\mathbb{E}_q[(w^{\text{seq}})^2]$) is $(1 + \sigma_r^2)^{-L}$ times the original number of samples. For $L = 500$:

σ_r	0.02	0.05	0.1
Second moment	1.22	3.48	145
Ratio ESS, share of the samples	about 82%	about 29%	less than 1%

So whether the exact ratio is usable depends on the accumulated mismatch, measured by the accumulated second moment or by D_2 : with a somewhat large mismatch and long responses, its variance quickly makes it unusable.

7.3 What is done in practice

GSPO’s geometric mean. GSPO [13] uses the $1/L_i$ -th power of the whole-response ratio, $r_i^{\text{geo}} = (p(y_i | x)/q(y_i | x))^{1/L_i}$, as the weight of the i -th response, and clips at the sequence level. This weight depends on the whole response but is not the correction of Eq. (35): the $1/L_i$ -th power keeps it close to 1, precisely to avoid a huge variance, so it no longer corresponds exactly to any one term. Its motivation is training stability, not unbiasedness. Ignoring clipping,

$$\nabla_{\theta} r_i^{\text{geo}} = r_i^{\text{geo}} \frac{1}{L_i} \sum_{t=1}^{L_i} g_{it},$$

where g_{it} is the score of the t -th token of the i -th response; the gradient formula in §4.2 of the GSPO paper writes out this relation [13]. So GSPO also averages the token scores of each response, with the same $1/L_i$ as the original GRPO, and the length analysis of Chapter 8 applies to it. In the umbrella formula it is a weight outside Premise P, counted in D.

Controlling the distance. Limiting the sample age and adding a KL penalty toward the sampler both keep p close to q , and so keep C small (Chapter 10). Another option is to estimate the trainer’s own values with a critic.

7.4 At the level of the objective, the two parts combine into one term

Let $\eta(\pi)$ be the expected return of policy π , γ the discount, and d_{π} the distribution of the prefixes that π visits (the discounted visit frequencies, normalized to a distribution). The performance-difference lemma [21] says

$$\eta(\pi) - \eta(q) = \frac{1}{1 - \gamma} \mathbb{E}_{s \sim d_{\pi}} \mathbb{E}_{a \sim \pi} [A^q(s, a)].$$

So for the **value** of the objective, the sampler’s advantage is exact as long as the prefixes come from the right policy: at the level of values, C2 does not appear on its own, only together with C1.

The TRPO surrogate $L_q(\pi)$ uses the prefixes of q , and $\eta(\pi) - L_q(\pi)$ is the value-level counterpart of C (§3.3 showed that $C = \nabla_{\theta} [L_q(p_{\theta}) - J(p_{\theta})]$). The two functions agree to first order at $\pi = q$, so C is second order for the objective and first order for the gradient. This “second order” is in the square of the total variation distance (the TRPO bound [22], Chapter 11), and unlike the KL bound for B (§6) it also holds at the boundary of the support.

C also grows with length. The prefix of the t -th token consists of the $t - 1$ choices made before it, and the KL between the sampler’s and the trainer’s distributions over these prefixes is the sum of the expected per-token KL over those steps, so it grows roughly linearly in t . In the TRPO-style bound this term has a factor $1/(1 - \gamma)^2$, one more factor of the horizon length than B has (Chapter 11).

Cells of Table 1 filled in this chapter: The whole-response ratio row and the GSPO row.

What remains after this term: A, B, and C all assume Premise P. Methods used in practice violate it in a few places (GRPO’s $1/L_i$, the group standard deviation, the length-weighted baseline); the next chapter computes what this costs, also on-policy.

8 Term D: methods outside Premise P

$$\mathbb{E}\|\hat{g}_N - \nabla_{\theta} J\|^2 = \left\| A + B + C [+ \textcolor{red}{D}] \right\|^2 + \frac{\text{tr Cov}(\hat{g})}{N}$$

This chapter answers: What happens when the baseline looks at a response’s own reward, or the weight at its length, and how do we measure the resulting length bias? Where do GRPO’s $1/L_i$, the group standard deviation, the length-weighted baseline, DAPO, and Dr. GRPO fit?

These effects do not come from $q \neq p$, so we work on-policy. Here $D = \mathbb{E}[\hat{g}] - \mathbb{E}[\hat{g}_0]$ for a reference estimator $\hat{g}_0$ that satisfies Premise P (§3.3); §8.8 lists $\hat{g}_0$ per method and treats D off-policy.

8.1 One principle: the coefficient of a response gives the local surrogate of the current update

Collect everything that multiplies the tokens of the i -th response into one coefficient c_i :

$$\hat{g} = \sum_i c_i S_i, \quad S_i = \sum_t g(s_{it}, y_{it}) = \nabla_{\theta} \log p(y_i | x). \quad (36)$$

For GRPO [10] (without dividing by the standard deviation), $c_i = (R_i - \bar{R})/(nL_i)$; for DAPO [11], $c_i = (R_i - \bar{R})/\sum_j L_j$; for Dr. GRPO [12], $c_i = (R_i - \bar{R})/C$ with a fixed constant C .

The scalar c_i (not the centering vector $c(s)$ of Chapter 1) also depends on the other responses in the group. At the current parameters θ_0 , we freeze their distribution and define

$$\bar{c}(y; \theta_0) = \mathbb{E}_{\text{other responses} \sim p_{\theta_0}} [c_i(y; \text{other responses})], \quad \mathcal{F}(\theta; \theta_0) = \mathbb{E}_{y \sim p_{\theta}} [\bar{c}(y; \theta_0)].$$

If c_i depends on θ only through the responses, the score identity (Appendix A.2) gives

$$\mathbb{E}[c_i S_i] = \nabla_{\theta} \mathcal{F}(\theta; \theta_0) \Big|_{\theta=\theta_0}. \quad (37)$$

So **the current update is the gradient of the local surrogate** $\mathcal{F}(\cdot; \theta_0)$: near θ_0 , the policy moves in the direction that makes $\bar{c}$ larger.

How to check whether a method has a length bias

1. Freeze the other responses and write c_i as a function of this response’s reward R_i and length L_i . If c_i is a positive constant times R_i minus a quantity that does not involve this response, the update is a positive multiple of $\nabla_{\theta} J$; any other dependence on this response is gradient ascent on a reward you never wrote down, and this is where D comes from.
2. With R_i fixed, any dependence of c_i on L_i is a length preference the reward did not ask for, measured by the **length elasticity** $\partial \log |c_i| / \partial \log L_i$ (the percent change of the weight when the response is 1% longer) or the **implied per-token cost** $\partial c_i / \partial L_i$ (in the units of a per-token length penalty; negative is a cost, positive a bonus).

The local surrogate is not a global objective: $\bar{c}$ changes with θ_0 through the distribution of the other responses, and some such updates are not the gradient of any fixed function (Appendix A.13).

8.2 Applying the principle to different methods

Write $c_i = (R_i - b) \ell_i$, where b does not involve this response and $\ell_i = \ell(L_i) > 0$ is a length factor (unrelated to the SFC score h). After freezing, b is a number with mean b_0 , so by Eq. (37) the local surrogate is $\mathbb{E}_{p_\theta}[(R - b_0) \ell(L)]$. A constant ℓ (Dr. GRPO, RLOO) gives no length preference; an ℓ that decreases with L prefers short responses when $R > b$ and long ones when $R < b$.

GRPO: $\ell = 1/L$. The local surrogate is $\mathbb{E}[(R - b_0)/L]$, so wrong responses keep getting longer (as Dr. GRPO pointed out); doubling the length halves the weight (elasticity -1) for any n .

DAPO’s per-group normalization. This response’s length is in the denominator: $\ell_i = 1/(L_i + \Lambda_i)$ with $\Lambda_i = \sum_{j \neq i} L_j$, so the elasticity is $-L_i/(L_i + \Lambda_i)$, about $-1/n$ for a typical response. For a 0/1 reward the conclusions above hold although ℓ_i also depends on the other lengths. With the usual whole-batch average, Λ_i is very large and the effect negligible.

The length-weighted baseline (LW). OPO [14] uses it to approximate the minimum-variance baseline (15). The length enters through the baseline $b_{\text{LW}} = \sum_j L_j R_j / \sum_j L_j$, which gives a response’s own reward the weight $\omega_i = L_i / \sum_j L_j$. By the self-inclusion rule (12),

$$R_i - b_{\text{LW}} = (1 - \omega_i)(R_i - b_{-i}^{\text{LW}}), \quad \omega_i = \frac{L_i}{L_i + \Lambda_i}, \quad b_{-i}^{\text{LW}} = \frac{\sum_{j \neq i} L_j R_j}{\sum_{j \neq i} L_j}. \quad (38)$$

So $\ell_i = 1 - \omega_i = \Lambda_i / (L_i + \Lambda_i)$, and the elasticity $-\omega_i$ is larger in magnitude for a response with a large share of the tokens. The leave-one-out version uses b_{-i}^{LW} , so $\ell = 1$.

Method	Length factor $\ell(L_i)$	Elasticity	Effect
GRPO	$1/L_i$	-1	Strong; does not shrink with n
DAPO, per-group normalization	$1/(L_i + \Lambda_i)$	$-\omega_i \approx -1/n$	Weak
DAPO, per-batch normalization	Nearly constant	≈ 0	Negligible
Dr. GRPO (constant divisor)	Constant	0	None
RLOO, token sum	Constant	0	None
Length-weighted (self-included)	$\Lambda_i / (L_i + \Lambda_i) = 1 - \omega_i$	$-\omega_i \approx -1/n$	Weak
Leave-one-out length-weighted	Constant	0	None

8.3 The same thing, viewed as an average over the group

For a token sum and one baseline $\bar{b}$ for the whole group, an exact identity holds (Appendix A.12):

$$\mathbb{E}\left[\sum_{i=1}^n (R_i - \bar{b}) S_i\right] = n \nabla_{\theta} J - \nabla_{\theta} \mathbb{E}[\bar{b}]. \quad (39)$$

For the group mean, $\mathbb{E}[\bar{R}] = J$, so the expected update is $(n-1)\nabla_{\theta} J$. For the length-weighted baseline, $b_{\text{LW}} = \bar{R} + \text{Cov}_G(L, R)/\bar{L}$ (within-group covariance Cov_G , mean length $\bar{L}$), so

$$\mathbb{E}[\hat{g}_{\text{LW}}] = (n-1) \nabla_{\theta} J - \nabla_{\theta} \mathbb{E}\left[\frac{\text{Cov}_G(L, R)}{\bar{L}}\right]. \quad (40)$$

The second term lowers the within-group covariance of length and reward, making correct responses shorter relative to wrong ones. In a large group (moments and derivatives under control) it is of order $1/n$ relative to the main term, but along directions that change the length and not the accuracy, $\nabla_{\theta} J$ is zero and, apart from an explicit length penalty, it is the only nonzero gradient. For a 0/1 reward with success rate ϱ and mean lengths L_c of correct and L_w of wrong responses, its large-group limit is $\text{Cov}(L, R)/\mathbb{E}[L] = \varrho(1-\varrho)(L_c - L_w)/(\varrho L_c + (1-\varrho)L_w)$.

8.4 Two worked examples

Two responses of very different lengths. With $L_1 = 1000$, $L_2 = 1$, and the long response correct, $b_{\text{LW}} = 1000/1001 \approx 0.999$: the long one gets +0.001 per token instead of the leave-one-out +1, and with rewards swapped −0.001 instead of −1, the same direction as GRPO’s $1/L$ bias.

Four responses							
Response	R_i	L_i	ω_i	Group mean	LW	Leave-one-out LW	$1 - \omega_i$
A	1	2,000	0.1	+0.5	+0.6	+0.667	0.9
B	1	6,000	0.3	+0.5	+0.6	+0.857	0.7
C	0	10,000	0.5	−0.5	−0.4	−0.800	0.5
D	0	2,000	0.1	−0.5	−0.4	−0.444	0.9
Token-weighted sum $\sum_i L_i A_i$				−2000	0	−2413	

The LW column is the leave-one-out column times $1 - \omega_i$, so the long wrong response C, with half of the tokens, keeps half of its negative signal per token. The net push (last row) shows the trade-off: with unequal lengths, zero net push and no self-inclusion bias cannot both hold.

A three-parameter toy model [toy]. The policy has logits a for the success rate and β_c , β_w for a correct and a wrong response to be long (length 800 instead of 200); the reward looks only at correctness, so the update along β_c and β_w is entirely bias. At $a = \beta_c = \beta_w = 0$ (exact sums over all groups), its ratio to the update along a is 0.42 for GRPO’s $1/L_i$ for every n from 2 to 16, about $0.44/n$ for the length-weighted baseline and, for large n , about $0.42/(n-1)$ for DAPO’s per-group normalization, and exactly zero for the constant divisor, RLOO, and leave-one-out LW.

8.5 The group standard deviation

GRPO also divides by the group standard deviation, which depends on R_i too; §4.2 gives the conditional mean $\chi(R)$. For a reward with two values, χ is linear: within one prompt it is a fixed baseline times a constant ζ , a positive scaling that keeps the direction. But ζ changes with the success rate (for $n = 8$, about 1.86 at success rate 0.5 and about 2.11 at 0.2 or 0.8), so prompts with extreme success rates get more weight: the difficulty bias that Dr. GRPO describes.

With three or more values, χ can be nonlinear: for $n = 2$ and a reward that independently takes each of the values $\{0, 1, 10\}$ with equal probability, the signal is $\text{sign}(R_i - R_j)$, so $\chi = (-2/3, 0, 2/3)$, which is not proportional to $R - \mathbb{E}[R] = (-11/3, -8/3, 19/3)$, and the local surrogate is $\mathbb{E}[\chi(R)]$ rather than $\mathbb{E}[R]$. Both effects change the objective; neither is a length effect.

8.6 Clipping and masks that depend on the sign of the signal

Clipping in the style of PPO [23] (including DAPO’s clip-higher [11]) gives zero gradient weight to tokens whose ratio is above the upper limit when the signal is positive, and below the lower limit when it is negative. The weight then depends on the sign of $R - b$, which violates (P2). This is a deliberate change of the objective, to limit how much one step changes the policy.

With ratio-only truncation or masking (TIS, MIS), we can weight first and then center, and the expected update no longer depends on the baseline (§5.4). Not with the PPO mask: changing the baseline changes the sign of the signal, and so which weights are used. Example: two tokens, $q = (0.1, 0.9)$, $p = (0.2, 0.8)$, and the parameter is the log-odds of token 1, so $g = (0.8, -0.2)$ and $r = (2, 8/9)$; the clip range is $[0.8, 1.2]$. A positive signal masks token 1, so $w_+ = (0, 8/9)$; a negative one masks ratios below 0.8, so none: $w_- = (2, 8/9)$. Exact re-centering gives $f_+ = w_+g - \mathbb{E}_q[w_+g] = (0.16, -0.0178)$ and $f_- = w_-g - \mathbb{E}_q[w_-g] = (1.6, -0.178)$, both with mean zero under q .

With deterministic returns $R = (1, 0)$ and $b = 0$, token 1 has a positive signal and token 2 a zero signal, so the expected update is $0.1 \times 1 \times 0.16 = 0.016$. With $b = 0.5$, token 1 uses f_+ and token 2, now negative, uses f_- , giving $0.1 \times 0.5 \times 0.16 + 0.9 \times (-0.5) \times (-0.178) = 0.088$. Each set has mean zero under the full q , but the set actually used is picked by the sign of each token’s signal, and multipliers picked this way no longer have mean zero.

The public code of Marek and Ryabinin combines centering with this kind of sign mask, taking the centering weights from one of the two sets by the sign of the signal (`log_mass` in `losses.py`) [2]; by the example above, its expected update still depends on the baseline. So the PPO mask is left out of the baseline-invariance guarantee of Chapter 5 and counted in D. Centering can still be used with it, but the actual estimator and its guarantees must then be written out again.

8.7 Common misreadings

“With a token-level loss, the group mean’s negative net push favors short responses.” Not on-policy: with a constant divisor the expected update is exactly $(n-1)\nabla_\theta J$ (Eq. (39)); the net push multiplies a score with mean zero and only adds noise. It is a bias only off-policy (Chapter 4).

8.8 The reference estimator for each method, and D off-policy

The table lists $\hat{g}_0$ for each method. When a method violates P in several places, $\hat{g}_0$ replaces all of them, and D is the total difference, which need not split into one difference per place. In the last two rows $\hat{g}_0$ is centered, so D also counts the drift that the actual method does not remove.

Method	Where P is violated	Reference $\hat{g}_0$	What D is
Self-included group mean (as in GRPO, Dr. GRPO)	Own reward in the baseline (P1)	Leave-one-out, same normalization	Sample by sample, $\hat{g} = (1 - \frac{1}{n})\hat{g}_0$, so $D = -\frac{1}{n}\mathbb{E}[\hat{g}_0]$: only a scaling
GRPO's $1/L_i$	Own length in the weight (P2)	Leave-one-out, constant divisor	An extra factor $1/L$ in the local surrogate (plus a scaling by $1 - \frac{1}{n}$): a length preference
DAPO's per-group normalization	Own length in the divisor (P2)	Leave-one-out, constant divisor	A length preference with elasticity about $-\omega_i$ (plus a scaling by $1 - \frac{1}{n}$)
Self-included length-weighted baseline (token sum)	Own reward and length in the baseline (P1)	Leave-one-out length-weighted	On-policy, $D = -\nabla_{\theta} J - \nabla_{\theta} \mathbb{E}[\text{Cov}_G(L, R)/\bar{L}]$: a scaling plus a length term
Group standard deviation	Own reward in the signal (P1)	Leave-one-out, no division by the standard deviation	The reward is replaced by $\chi(R)$: a per-prompt scaling for a binary reward (difficulty bias), a changed objective for more values
PPO's sign mask	Sign of the signal in the weight (P2)	Ratio truncation, re-centered	The expected update depends on the baseline (§8.6)
GSPO	Whole response in the weight (P2)	Per-token ratio truncation, re-centered	A sequence-level weight, plus the length factor $1/L_i$ (§7.3)

Off-policy. For a baseline that depends on this response, Eq. (22) gives

$$D = - \sum_t \mathbb{E}_{s_t \sim q} [\text{Cov}(b, f \mid s_t)],$$

with the reference baseline of §3.3. This covariance remains when SFC or IS makes $\bar{f}$ zero. A length factor acts the same way: in the signal $(R-b)\ell(L)$, the part $b\ell(L)$ depends on this response's length. So D does not go away together with A (item 3 of §5.6); removing it requires a different baseline or normalization.

Cells of Table 1 filled in this chapter: D column: length weighting (self-inclusion bias, about $O(1/n)$ in large groups); GRPO's $1/L_i$ and group standard deviation (surrogate changed); RLOO and GRPO group mean (only a scaling); PPO's sign mask (baseline-dependent update).

What remains after this term: That ends the four bias terms. The umbrella formula's other half is the variance: each fix for A or B adds some, and the next chapter computes how much.

Part III Variance, choices, and guarantees

9 Variance: the cost of fixing the bias

$$\mathbb{E}\|\hat{g}_N - \nabla_{\theta} J\|^2 = \left\| \underbrace{A}_{\text{source 1}} + \underbrace{B}_{\text{source 2}} + \underbrace{C}_{\text{source 3}} + \underbrace{D}_{\text{size of the multiplier}} \right\|^2 + \frac{\text{tr Cov}(\hat{g})}{N}$$

This chapter answers: How large is each kind of noise in a finite batch, and how do the baseline, the ratio, and the alternative tokens change it? When does variance matter more than bias?

9.1 Starting from the per-prefix formula

By Eq. (23) of §3.8, at a prefix s the second moment of the update $(R - b)f_{\xi}$ from a single token is

$$\sum_v q(v) \left[\underbrace{(A^q(v) + \bar{A}_q)^2}_{\text{source 1}} + \underbrace{\sigma^2(v)}_{\text{source 2}} + \underbrace{\text{Var}(b | s)}_{\text{source 3}} \right] \underbrace{\mathbb{E}_{\xi} \|f_{\xi}(v)\|^2}_{\text{size of the multiplier}},$$

and the variance is this minus the squared mean. Averaging N independent samples keeps the bias and divides the variance by N , so the mean squared error (MSE) with respect to the target is

$$\text{MSE}(N) = \|\text{bias}\|^2 + \frac{\text{tr Cov}(\hat{g})}{N}. \quad (41)$$

What N counts. Responses that serve as each other's baselines are not independent, even if the rollouts are. The smallest example is on-policy: two actions $a \in \{0, 1\}$, $p(a = 1) = 1/2$, return $R = a$, log-odds score $g(a) = a - 1/2$, and two responses, each the other's leave-one-out baseline. The updates $X_1 = (R_1 - R_2)g(a_1)$ and $X_2 = (R_2 - R_1)g(a_2)$ are both 0 when the actions agree and both $1/2$ when they differ, so $X_1 = X_2$, and their average still has variance $1/16$, not $1/32$.

So in this note a **group of independent responses** is one sampling unit, and N counts groups. Counting responses requires the cross terms, $\text{tr Cov}(\frac{1}{N} \sum_i X_i) = \frac{1}{N^2} \sum_{i,j} \text{tr Cov}(X_i, X_j)$, which are exact for on-policy leave-one-out (Appendix A.15): with $X_i = (R_i - \bar{R}_{-i})S_i$ in a group of n , $\text{Cov}(X_i, X_j) = \nabla_{\theta} J \nabla_{\theta} J^{\top} / (n-1)^2$ for $i \neq j$, so

$$\text{tr Cov}\left(\frac{1}{n} \sum_{i=1}^n X_i\right) = \frac{\text{tr Cov}(X)}{n} + \frac{\|\nabla_{\theta} J\|^2}{n(n-1)}.$$

The second term is $1/((n-1)N_1^*)$ times the first, where $N_1^* = \text{tr Cov}(X)/\|\nabla_{\theta} J\|^2$ is the N^* of Eq. (14) for single responses: the two terms are equal in the two-action example ($N_1^* = 1$, $n = 2$), and the correction is negligible when N_1^* is in the hundreds or thousands. Off-policy, or with other baselines, the cross terms must be computed separately.

The examples below use one prefix, independent main samples, a deterministic return ($R = Q^q(v)$, so source 2 is zero), and a fixed baseline (source 3 is zero); their $N^* = \text{tr Cov}(\hat{g})/\|\mathbb{E}\hat{g}\|^2$ compares methods and is not the number of responses a real batch needs, still less a number of tokens.

9.2 Source 1: the common offset, set by the baseline

The $\bar{A}_q^2$ in source 1 is what the baseline leaves behind. Once SFC makes A zero, this term is where the baseline matters (its fluctuation also enters source 3, and through $\mathbb{E}[A^2]$ it enters V_K , §9.4).

Example 1 (off-policy): N^* of SFC for each baseline and K

$q = (0.6, 0.4)$, $Q^q = (0.91, 0.89)$, $h = (0.4, -0.6)$; the expected update is 0.0048 for every b . “Exact centering”: m_q known; K : m_q estimated with K alternative tokens, adding V_K (§9.4).

	Values of one sample	Exact centering	$K = 4$	$K = 1$
$b = 0$	0.364, -0.534	8400	10519	16876
$b = 0.5$ (the prompt’s success rate)	0.164, -0.234	1650	2071	3334
$b = 0.902 = V^q$	0.0032, 0.0072	0.17	0.42	1.17

SFC removes the shared part of the signal (about 0.4 at $b = 0.5$) from the **expectation**, not the **variance**, so the baseline still matters; at $b = V^q$ one sample has the right sign.

The variance-minimizing baseline, and the cost of an inaccurate one. With exact centering, the expected update of SFC does not depend on b , so the best b minimizes $\mathbb{E}_q[(R - b)^2 \|h\|^2]$:

$$b_{\text{var}} = \frac{\mathbb{E}_q[R \|h\|^2]}{\mathbb{E}_q[\|h\|^2]}, \quad \mathbb{E}_q[(R - b)^2 \|h\|^2] = \mathbb{E}_q[(R - b_{\text{var}})^2 \|h\|^2] + (b - b_{\text{var}})^2 \mathbb{E}_q[\|h\|^2]. \quad (42)$$

The second equation is exact: any other b adds exactly $(b - b_{\text{var}})^2 \mathbb{E}_q[\|h\|^2]$ of variance and no bias, which makes the statement of §2.4 exact. In Example 1, $b_{\text{var}} = 0.898$, close to $V^q = 0.902$; with it the variance is zero, which, as in Chapter 2, is special to two tokens and one parameter.

9.3 Sources 2 and 3

Source 2: return noise from the continuation. It is also multiplied by $\|f\|^2$, so its cost depends on the method. With σ_v^2 the variance of the return after token v , in Example 2 ($q = (0.1, 0.1, 0.8)$, $p = (0.8, 0.1, 0.1)$) SFC with exact centering adds $\sum_v q(v) \sigma_v^2 \|h(v)\|^2 = 0.146 \sigma_1^2 + 0.146 \sigma_2^2 + 0.048 \sigma_3^2$, and exact IS adds $\sum_v q(v) r(v)^2 \sigma_v^2 \|g(v)\|^2 = 0.384 \sigma_1^2 + 0.146 \sigma_2^2 + 0.018 \sigma_3^2$: much more on token 1, which the sampler rarely draws and the trainer often picks (§9.4).

Source 3: fluctuation of the baseline. The leave-one-out average of the other $n - 1$ responses is independent of this response; its variance $\text{Var}(R)/(n - 1)$, with $\text{Var}(R)$ the return variance of one rollout for the prompt, is multiplied by $\mathbb{E}_q \mathbb{E}_\xi \|f_\xi\|^2$ (for the group mean, scale by $1 - 1/n$). It shrinks as n grows, is usually much smaller than source 1, and under Premise (P1) adds no bias.

9.4 The size of the multiplier: what the weight and the centering add

The weight w : the effective sample size of exact IS. When $f = rg$, $\|f\|^2 = r^2 \|g\|^2$. Assuming support coverage, the ratio has mean $\mathbb{E}_q[r] = \sum_v p(v) = 1$ and variance

$$\text{Var}_q(r) = \mathbb{E}_q[r^2] - 1, \quad \mathbb{E}_q[r^2] = \sum_v \frac{p(v)^2}{q(v)} = \exp(D_2(p \| q)), \quad (43)$$

where D_2 is the Rényi divergence of order 2. The usual **effective sample size** (ESS) of N ratios, $(\sum_i r_i)^2 / \sum_i r_i^2$, is N when all the ratios are equal and close to 1 when one dominates. Its numerator is about $(N \mathbb{E}_q[r])^2 = N^2$ and its denominator about $N \mathbb{E}_q[r^2]$, so $\text{ESS} \approx N / \mathbb{E}_q[r^2]$.

The ESS measures how concentrated the ratios are, not directly how much the variance grows. For the averaged quantity $X = (R - b)g$, with $m = \mathbb{E}_p[X] = \mathbb{E}_q[rX]$, the exact variance ratio is

$$\frac{\text{tr Cov}_q(rX)}{\text{tr Cov}_p(X)} = \frac{\mathbb{E}_q[r^2\|X\|^2] - \|m\|^2}{\mathbb{E}_p[\|X\|^2] - \|m\|^2},$$

close to $\mathbb{E}_q[r^2]$ only when $\|X\|^2$ is about constant across tokens and $\|m\|^2$ is negligible. Counterexample: $p = (0.99, 0.01)$, $q = (0.5, 0.5)$, $X = (1, -1)$, so $\|X\|^2$ is 1 everywhere and $\mathbb{E}_q[r^2] = 1.9604$, but the on-policy variance is $1 - 0.98^2 = 0.0396$ and the weighted variance $1.9604 - 0.98^2 = 1$, a ratio of about 25.25. In Example 2, $\mathbb{E}_q[r^2] = 6.5$ (ESS about 15% of the samples), but N^* is 11.9 for exact IS and 1.44 on-policy from p : a ratio of 8.3, the same order as 6.5, not equal.

Right on average, but almost never right in a single batch

The trainer is $p = (0.5, 0.5)$ and the sampler is $q = (0.999, 0.001)$. From 100 samples of the sampler we estimate the trainer’s probability of the rare second token, 0.5, by $\frac{1}{100} \sum_i r(v_i) \mathbf{1}\{v_i = 2\}$. The rare token’s ratio is $0.5/0.001 = 500$, so the estimate is $5 \times$ (its count in the batch):

Times the rare token appears	0	1	2
Probability	90.5%	9.1%	0.45%
Estimate	0	5	10

The estimate is exactly 0.5 on average, but a single batch almost never gives a number close to it; $\mathbb{E}_q[r^2] \approx 250$, so the ESS of the 100 samples is below 1.

Example 2 is milder: token 3 is 80% of the samples, but with ratio 0.125 its update $(0.015, 0.002, -0.017)$ is almost zero, while token 1, only 10% of the samples, has ratio 8 and update $(1.36, -0.68, -0.68)$ and gives almost all of it; a batch of 10 has no token 1 with probability $0.9^{10} \approx 35\%$. D_2 reacts to rare tokens much more strongly than the KL, so this cost can be large while the monitored KL is small; it is infinite when the sampler truncates but the trainer does not.

The centering c : V_K , **the cost of estimating the average.** SFC estimates m_q with K alternative tokens (Eq. (29)), which must be

- distributed like the main token (same temperature and truncation), so $\mathbb{E}[\nabla_\theta z(a'_k)] = m_q$;
- drawn with replacement, since forbidding repeats changes the distribution: with $q = (0.9, 0.1)$, one alternative token that must differ from the main token has distribution $(0.1, 0.9)$;
- independent of the main token’s outcome.

Writing the update from one main sample as the exactly centered update minus an error,

$$\hat{X} = A \hat{h}(a) = A \underbrace{(\nabla_\theta z(a) - m_q)}_{X_0} - A e, \quad e = \hat{m}_q - m_q,$$

the three requirements give $\mathbb{E}[e] = 0$, e independent of the main token and its signal, and $\text{Cov}(e) = \Sigma_g/K$, where $\Sigma_g = \text{Cov}_{v \sim q}(\nabla_\theta z(v))$ is the covariance of the logit gradient of one alternative token.

So the mean does not change, and the variance grows by (Appendix A.14)

$$V_K = \frac{\mathbb{E}[A^2] \operatorname{tr} \Sigma_g}{K}, \quad (44)$$

which is the term $\operatorname{tr} \operatorname{Cov}_\xi(f_\xi) = \operatorname{tr} \Sigma_g / K$ of Eq. (23) times the second moment of the signal. In Example 1 with $b = 0.5$ and $K = 4$: $\operatorname{tr} \Sigma_g = 0.6 \times 0.4 = 0.24$, $\mathbb{E}[A^2] = 0.6 \times 0.41^2 + 0.4 \times 0.39^2 = 0.162$, and $V_K = 0.162 \times 0.24 / 4 = 0.0097$, matching a Monte Carlo average of 0.0097 over 20 random seeds of 2×10^5 samples (standard deviation 0.00008 across seeds). Three consequences:

- A baseline that centers the signal also lowers V_K , because $\mathbb{E}[A^2] = \operatorname{Var}(A) + \bar{A}_q^2$.
- On-policy, $m_q = m_p$ is known from backpropagation, so alternative tokens only add V_K .
- With the head computed exactly from the logged top- k probabilities and only the tail sampled, Σ_g becomes $q(\bar{H})^2 \operatorname{Cov}_{q(\cdot|\bar{H})}(\nabla_\theta z)$, much smaller for a peaked distribution (§C.1).

In one sentence: V_K is the squared signal times the spread of the logit gradient under the sampler, over K : no ratio, so no $1/q$ amplification, but it grows with the signal.

9.5 Both sample from q : why the sources of variance differ

Both sample from q ; with a deterministic return and a fixed baseline, $A(v) = Q^q(v) - b$, their single-sample second moments differ only in what each sample is multiplied by:

$$\begin{aligned} \text{Exact IS: } & \sum_v q(v) r(v)^2 A(v)^2 \|g(v)\|^2 = \sum_v \frac{p(v)^2}{q(v)} A(v)^2 \|g(v)\|^2, \\ \text{SFC: } & \sum_v q(v) A(v)^2 \|h(v)\|^2 + V_K, \end{aligned}$$

and in general $A(v)^2$ is replaced by $\mathbb{E}[(R - b)^2 \mid s, v]$. In exact IS, $q(v)$ is in the **denominator**, so the rare tokens of §9.4 dominate, without bound as $q(v) \rightarrow 0$. In SFC, $q(v)$ is in the **numerator**, and V_K has no ratio (it depends on the spread of $\nabla_\theta z$, set for a neural network by the Jacobian of the logits). This is a trade between bias and variance: giving such a token its share under p takes large weights on a few samples, the variance of IS; SFC avoids it by keeping the weights at q , which is term B.

9.6 Putting bias and variance together

(a) SFC versus exact IS (Example 2, with $G[p]$ as the target). SFC has term B, with $\|\text{bias}\|^2 = 0.0074$ and single-sample variance 0.102; exact IS has no term B and variance 0.273 (both with the baseline V^q and the exact average). By Eq. (41):

Samples at the same prefix, N	1	5	10	23	50	100
MSE of SFC	0.109	0.028	0.018	0.0118	0.0094	0.0084
MSE of exact IS	0.273	0.055	0.027	0.0119	0.0055	0.0027

The crossover is at about 23 samples (about 32 with a baseline of 0). In real training a prefix usually appears once, but the gradient sums many prefixes: if B points in correlated directions on different prefixes it adds up while independent noise averages out, which favors IS; heavy-tailed ratios let a few tokens dominate the sum, which works against IS. So the effective sample size decides.

(b) TIS with re-centering: between the two [toy].

- **Setup.** Random tabular problems with 20 tokens: sampler logits from $\mathcal{N}(0, 1.5^2)$; $Q^q(v)$ uniform on $[0, 1]$; trainer logits $z_q + \kappa A^q$ (moved along the advantage) plus independent normal noise per token with standard deviation 0.2κ , so a larger κ means a larger lag. Ratio cap $\bar{r} = 2$; 3000 problems per κ ; random seed 7.
- **What is compared.** With $G[p]$ as the reference target: $G[q]$ for SFC, $G[\min(p, \bar{r}q)]$ for TIS with re-centering (“TIS” in the table), and $G[p]$ for exact IS. Variances are of the single-sample update (signal A^q , exact centering, no V_K), as median multiples of SFC’s.
- **Measures.** ESS: median of $1/\mathbb{E}_q[r^2]$. $\bar{w} = \mathbb{E}_q[\tilde{w}]$: average total mass of the weights. Gap ratio: $\|G[p] - G[\min(p, \bar{r}q)]\| / \|G[p] - G[q]\|$, below 1 if closer to the target than SFC. Rescaled: truncated update divided by $\bar{w}$. Cosine: median cosine of an expected update with $G[p]$.

κ	ESS	$\bar{w}$	Median gap ratio (fraction < 1)		Cosine with $G[p]$		Variance (multiple of SFC)		
			Raw	Rescaled	SFC	TIS	TIS, raw	TIS, rescaled	Exact IS
1	0.92	1.00	0.00 (100%)	0.00 (100%)	0.98	1.00	1.16	1.17	1.17
2	0.73	0.95	0.20 (99%)	0.19 (100%)	0.91	1.00	1.33	1.46	1.70
4	0.41	0.78	0.50 (90%)	0.49 (89%)	0.68	0.90	1.13	1.84	3.12
8	0.16	0.52	0.54 (90%)	0.78 (63%)	0.28	0.30	0.74	2.85	5.57

Observations. With a small lag, almost no token has $r > \bar{r}$, so truncation with re-centering is just exact IS. With a moderate lag ($\kappa = 2, 4$), it is closer to the target than SFC in about 90% of the cases or more, with a rescaled variance below that of exact IS. With a very large lag ($\kappa = 8$), little advantage is left once the scaling is removed (the raw variance is low partly because only half of the weight mass is left); SFC and truncation both have a cosine of only about 0.3 with $G[p]$, and exact IS has the right expected direction at 5.6 times the variance of SFC.

What this does not show. The toy has one prefix, a tabular parameterization, and exact centering: no term C, no V_K , no correlation between prefixes. It shows that at moderate lag TIS with re-centering can be a compromise between the two, not that it beats both in real training.

Evidence from training [empirical]: Marek and Ryabinin [1] (§5.3, Figure 4) report that for Qwen3-0.6B on Countdown, with a sampler refreshed only every 64 steps, SFC combined with TIS or MIS works better than SFC alone; their abstract adds that the additive correction can be combined with IS and that, in their staleness experiments, the combination beats IS alone. This agrees with the toy in direction, for one model, task, and refresh interval; such comparisons should align step sizes, or part of the difference is only one of learning rate.

(c) SFC versus plain REINFORCE: centering can also raise the variance. Two signals ($q(1) = 0.5$, $p(1) = 0.6$, scalar log-odds parameter, $g(a) = a - 0.6$, $h(a) = a - 0.5$, $K = 100$, and the expectation of SFC as the target):

Signal	$A(1) = 1, A(0) = 0$	$A(1) = 0.5, A(0) = -0.5$
Squared bias of plain REINFORCE	0.0025	0
Single-sample variance: plain / SFC (with V_K)	0.04 / 0.06375	0.0025 / 0.000625
Better at $N = 1$	Plain	SFC

In the first column, centering raises the base variance from 0.04 to 0.0625, and SFC wins only when $N > 9.5$. In the second, the signal is already centered ($\bar{A}_q = 0$) and neither estimator is biased; SFC's base variance is zero, so its 0.000625 is all V_K ($K = 100$). In general, centering changes the base variance by $\text{Var}(A)\|m_q - m_p\|^2 - 2(m_q - m_p)^\top \text{Cov}(Ag, A)$, whose sign the baseline decides.

(d) After many training steps [model]. If the bias points in correlated directions across steps, after M training steps (not N samples) it adds up roughly like $M\|\text{bias}\|$, and independent noise like $\sqrt{M \cdot \text{variance}}$, so a systematic bias matters more in long training. This rough rule assumes a fixed bias direction and noise independent across steps; neither holds exactly.

Cells of Table 1 filled in this chapter: The variance column: critic, RLOO, length weighting, SFC (+ V_K for a sampled center), exact IS (§9.4), TIS/MIS.

Next question: How do we choose given the target, why $q \neq p$, and the variance we can afford?

10 How to choose: target first, then source, then variance

This chapter answers: In real training, which terms are present, which need fixing, and with what? What does a KL penalty toward the sampler do?

Ask three questions in order, rather than starting from a method's name:

1. **Which policy do you want to improve?** The deployed sampler or the current trainer; this decides whether B and C are errors (§3.5).
2. **Why is $q \neq p$?** Engine mismatch, staleness, sampling settings, or several at once; this decides which terms are nonzero and which model applies.
3. **Can you afford the variance of the ratios?** The per-token effective sample size and D_2 (§9.4) decide between SFC, exact IS, and truncation.

10.1 Engine mismatch

The model. The rollout engine runs the current weights but computes the logits with an error, $q_\theta = \text{softmax}(z_\theta + \varepsilon)$, where $\varepsilon(v)$ may depend on the prefix and the token. The sampler's own score is (Appendix A.5)

$$\nabla_\theta \log q_\theta(v) = \underbrace{\nabla_\theta z(v) - m_q}_{h(v)} + \underbrace{\nabla_\theta \varepsilon(v) - \mathbb{E}_q[\nabla_\theta \varepsilon]}_{\text{present only if the error changes with } \theta}. \quad (45)$$

The **fixed-error model** of §3.5 sets $\nabla_\theta \varepsilon = 0$ (a stop-gradient on the error, the same construction as straight-through, STE; Appendix A.6), so $\nabla_\theta \log q_\theta = h$: SFC is exact REINFORCE for the

engine’s own policy [model].

Consequence. If deployment also uses this engine and the centering is accurate (the exact m_q or an unbiased estimate; a logged head with a modeled tail leaves the residual of §C.1), SFC gives exactly the gradient of the deployed q_θ , while ratios toward the trainer p would give the gradient of a policy that is not deployed, with a second moment about $e^{D_2(p||q)}$ times larger (§9.4). So SFC is exact here whatever the size of ε , while the cost of IS may grow with it: the cost depends on how much the differences in ε between tokens change the distribution, not on ε itself (adding the same large constant to all logits changes nothing).

This is consistent with the report that heavier quantization widens the gap between SFC and TIS (Marek and Ryabinin [1], Figure 1 and §5.4: for a 30B model whose sampler uses INT8 weights and activations and an INT4 KV cache, score centering reaches 30% training accuracy, TIS 12%, and every other method stays below 5%) [empirical]. It does **not** show that SFC becomes more exact with heavier quantization, which may make the error less smooth and more dependent on θ .

Where the model fails. The error can change with θ (mixture-of-experts routing depends on θ ; the rounding of hard quantization changes with the weights), and SFC then misses the second term of Eq. (45). Or deployment may use a third engine, and the target is neither q nor p .

10.2 Staleness

Here q is the trainer from a few updates ago and the target is the current trainer p , so all three terms are real errors.

- **A:** remove it with SFC, which is cheap and adds no bias of its own; its cost is V_K .
- **B:** negligible with a small lag and equal supports (§6); with a large lag, exact per-token IS or TIS with re-centering, depending on the ESS (§9.6 and the table below).
- **C:** with direct reweighting, it shrinks only if the lag stays small (sample-age limits, a KL penalty) or the full sequence-level ratio is used; it grows with the length of the response.

The bound of Chapter 11 guarantees only an improvement over the sampler, which under staleness may already be worse than the current trainer.

10.3 Different sampling settings

With a temperature or top- k /top- p truncation, q is not the trainer’s full softmax even with the same weights and engine. Compute q with the actual temperature and filtering rules (all that SFC needs), and handle the support as in §5.2: re-centering cannot recover tokens with $q = 0$, and a switch to a restricted target must be stated.

10.4 Several causes at once

Engine differences and lag often occur together. Two cheap measurements help: the KL between sampler and trainer with the same weights and zero lag (the engine part), and the same KL at the

normal lag. They cannot be subtracted, because the KL does not split into an engine part plus a lag part, but the first shows whether the engine difference alone is already non-negligible; the target must again be stated explicitly.

10.5 What a KL penalty toward the sampler does

By Eq. (24), adding $\beta D_{\text{KL}}(q||p_\theta)$ to the loss adds the ascent direction $\beta(m_q - m_p)$, the direction of the A term when $\bar{A}_q > 0$. Zhang [3] uses this term together with SFC, and the StableReinforce loss in Thinking Machines’ Tinker (currently in alpha) also adds such a KL term between the sampler and the learner, with a default coefficient of 0.01 [4].

- **With SFC:** the update is $G[q] + \beta(m_q - m_p)$ instead of $G[q] + \bar{A}_q(m_q - m_p)$ for plain REINFORCE: the coefficient $\bar{A}_q$ (set by the reward and the baseline, varying across prefixes, of either sign) becomes a positive β that you set. In the feedback-loop model of Appendix B.2 (λ the learning rate, τ the lag in steps, F the Fisher value along the update direction), the gain becomes $\lambda\tau\beta F > 0$: training can only slow down, never diverge [model].
- **Alone (without SFC):** the coefficient becomes $\bar{A}_q + \beta$, so the uncontrolled part remains.
- **As a trust region:** keeping p near q bounds B and C (the two bounds of §6, and Chapter 11); it limits the bounds, not the gap, because the KL does not see the advantage: policies differing only where the true advantage is near zero can have a large KL but a small gap.
- **With lag:** the trainer is a few steps ahead when a batch arrives, and the penalty moves it back toward the old policy, partly undoing those steps (more so for older samples); so the coefficient should decay with sample age: Zhang [3] uses $\beta_t = \beta_0/\sqrt{\text{age}_t}$ for sample age $\text{age}_t \geq 1$ (trainer updates since generation) and $\beta = 0$ for fresh samples.
- **With engine mismatch:** the KL comes mostly from the engine’s error, which the trainer cannot remove. With the sampler re-synchronized every step, in a toy with two actions and scalar log-odds θ , the extra term is $\beta[\text{sigmoid}(\theta + \varepsilon) - \text{sigmoid}(\theta)]$: independent of the reward, and nonzero whenever $\varepsilon \neq 0$ and the policy is not saturated [model].
- **Not a KL to a reference model:** the KL of most public recipes (for example GRPO [10]) is toward the starting SFT model and keeps the policy near its start; it has nothing to do with the sampler–trainer gap, and some newer recipes (for example DAPO [11]) drop it.

10.6 Decision table

Target; cause of $q \neq p$; ratios	Terms present	What to do	Why
All cases	A	First make the mean of the multiplier zero (SFC by default)	A is the gradient of an extra objective set by the baseline
Deployed engine; engine mismatch; error roughly fixed; accurate centering	Only A	SFC alone; do not add ratios toward the trainer	Under this model SFC is the gradient of the target

Target; cause of $q \neq p$; ratios	Terms present	What to do	Why
Current trainer; staleness, small lag	A; small B and C	SFC plus a limit on sample age (or a small KL penalty)	First order for the gradient, second order for the objective (for B, given the same support)
Current trainer; large lag; acceptable effective sample size	A, B, C	Exact per-token IS (SFC and IS correctly combined), plus an age limit	Fixes B; C still relies on the age limit
Current trainer; large lag; a few extreme ratios	A, B, C	TIS with re-centering, plus an age limit	Toy at moderate lag: median gap ratio to SFC about 0.2–0.5; weights never exceed $\bar{r}$
Current trainer; extreme ratios almost everywhere	A, B, C	Reduce the lag first	Per-token corrections are either too weak or too noisy
Sampling with truncation (top- k , top- p)	A, plus a support problem	Check the actual q and the trainer’s support; if needed, switch explicitly to a restricted target	Re-centering cannot recover tokens with $q = 0$
$1/L_i$, group standard deviation, or self-included length weighting	D	Switch to a constant divisor and leave-one-out	D does not go away together with A
A PPO-style sign mask	D	Analyze the actual estimator separately	Re-centering does not restore baseline invariance (§8.6)
Target or error model unclear	—	First specify the deployed policy and its gradient	The source’s label alone cannot tell whether there is a bias

10.7 Reading a recipe (to be verified by experiment)

Take one recipe: a length-weighted baseline, no division by the standard deviation, a token-level loss, SFC (K alternative tokens), no ratios, a KL term toward the sampler, and an explicit per-token length cost in the reward. For the current trainer as target and a small lag, the analysis and toys of this note say what each part does, not whether the recipe beats others.

- The length-weighted baseline includes the response itself, so it has a self-inclusion length bias (§8.2), $O(1/n)$ relative to the main term in large groups: its length elasticity of about $-1/n$ is a few percent to a little over ten percent for n from 8 to 16, and about one percent for n in the hundreds. In length-only directions, compare it with the explicit length cost.
- SFC removes A at both levels, so length weighting is left only to reduce the variance (§5.6).
- There are no ratios, so B and C are not fixed; the age limit and the KL term control them.
- When the explicit length cost is clipped away on failed rollouts, the small “per-token reward” from the self-inclusion bias is their only length signal; the leave-one-out length-weighted baseline removes it, but makes the zero net push only approximate (item 4 of §5.6).

Cells of Table 1 filled in this chapter: The row for a KL toward the sampler and limits on sample age (A, B, and C smaller, not zero).

Next question: What can we say about the **value** of the objective after a step, rather than the expectation and variance of one update? That is the TRPO-style bound.

11 The TRPO-style bound: which terms it controls and which it does not

This chapter answers: Is there a lower bound on the return after one step, and which terms of the umbrella formula does it control? Can it show that SFC is better than plain REINFORCE?

11.1 The bound

After a move to a candidate policy π , the return is at least the SFC surrogate minus two penalties, one for B and one for C. The notation:

Symbol	Meaning
$\eta(\pi), \gamma$	discounted return of policy π ; the discount
d_q	normalized discounted visit frequencies of the sampler's prefixes
$A_{\max}$	$\max_{s,a} A^q(s, a) $
$D_s, \bar{D}_{\text{KL}}(q\ \pi)$	$D_{\text{KL}}(q(\cdot s) \ \pi(\cdot s))$ at a single prefix; its average under d_q , $\mathbb{E}_{s \sim d_q}[D_s]$
α	the largest total variation distance, $\max_s \text{TV}(q(\cdot s), \pi(\cdot s))$, where $\text{TV}(q, \pi) = \frac{1}{2} \sum_a q(a) - \pi(a) $
C_{TR}	$4A_{\max}\gamma/(1 - \gamma)^2$

The SFC surrogate is

$$M_q^{\text{SFC}}(\pi) = \eta(q) + \frac{1}{1 - \gamma} \mathbb{E}_{s \sim d_q} \mathbb{E}_{a \sim q} \left[A^q(s, a) \log \frac{\pi(a | s)}{q(a | s)} \right],$$

and its gradient at $\pi = p_\theta$, prefix by prefix, is $\mathbb{E}_q[A^q g] = G_s[q]$, the SFC update. Combining the per-prefix inequality of Appendix A.7 with the TRPO bound [22] gives [identity]

$$\eta(\pi) \geq M_q^{\text{SFC}}(\pi) - \frac{A_{\max}}{1 - \gamma} \bar{D}_{\text{KL}}(q\|\pi) - C_{\text{TR}} \alpha^2. \quad (46)$$

The per-prefix step does not need the support of π to be covered by q . At the boundary of the support, however, the KL term is first order (§6), while the α^2 term is not affected.

What does the bound certify? Substitute the definition of M_q^{SFC} into Eq. (46). Only when the gain in the surrogate exceeds the sum of the two penalties,

$$\frac{1}{1 - \gamma} \mathbb{E}_{s \sim d_q, a \sim q} \left[A^q(s, a) \log \frac{\pi(a | s)}{q(a | s)} \right] > \frac{A_{\max}}{1 - \gamma} \bar{D}_{\text{KL}}(q\|\pi) + C_{\text{TR}} \alpha^2,$$

does the bound imply $\eta(\pi) > \eta(q)$. An increasing surrogate is not enough, and neither is a surrogate value higher than the current trainer's. Also, the comparison is with the sampler: under staleness, “better than the old sampler” does not mean “better than the current trainer”.

11.2 How the bound maps to the umbrella formula

- **B**: the term $A_{\max} \bar{D}_{\text{KL}}$ penalizes B at the level of the objective, that is, the difference between M_q^{SFC} (weights kept at q) and the TRPO surrogate L_q (weights changed to π) (Appendix A.7).
- **C**: the term $C_{\text{TR}} \alpha^2$ is the original penalty of TRPO [22], for the difference between $L_q(\pi)$ and $\eta(\pi)$, the value-level counterpart of C. It has an extra factor of $1/(1 - \gamma)$. Language models are usually not discounted ($\gamma = 1$); then one needs the finite-horizon version with length T , where the two factors correspond roughly to T and T^2 .
- **A**: does not appear. A is a property of the estimator, not of the return; the bound assumes that the surrogate is estimated accurately, and an estimator with an A term is just a wrong estimate of it.
- **Variance**: does not appear either. The bound says “if you reach policy π , the return is at least this much”; it does not say where finite samples will take you.

11.3 Which question it answers, and which it does not

“Is SFC better than plain REINFORCE?” contains three different questions:

1. **What is the lower bound on the return of a given candidate policy?** This is what the bound answers, and for the same candidate policy the two exact bounds, for SFC and for the plain surrogate, coincide. The plain surrogate uses the computed signal $A = A^q + \bar{A}_q(s)$, so it equals $M^{\text{SFC}} - \frac{1}{1-\gamma} \mathbb{E}_s[\bar{A}_q(s) D_s]$; its own penalty is $\frac{1}{1-\gamma} \mathbb{E}_s[(A_{\max} - \bar{A}_q(s)) D_s]$; substituting both gives Eq. (46) again.
2. **Which candidate policy will the training algorithm reach?** The A term changes the path of the optimization, and comparing the lower bounds of the same candidate policy cannot answer this.
3. **How well is the gradient estimated with finite samples?** This depends on the bias, the variance, the sample size, and the training dynamics (Chapter 9).

The real advantage of SFC here is that **its guarantee does not depend on the unknown offset in the signal**. Without a critic, $\bar{A}_q(s) = V^q(s) - \mathbb{E}[b \mid s]$ is unknown, so a usable guarantee for the plain surrogate must take the worst case over $|\bar{A}_q| \leq B_A$, at an extra cost of $\frac{1}{1-\gamma} \mathbb{E}_s[B_A D_s]$. The SFC surrogate does not need this step: when $\log(\pi/q)$ is centered under q , $\mathbb{E}_q[A(\log \frac{\pi}{q} - \mathbb{E}_q[\log \frac{\pi}{q}])] = \mathbb{E}_q[A^q \log \frac{\pi}{q}]$, so $\bar{A}_q$ cancels, and the surrogate can be estimated directly from the computed signal.

Next question: The next chapter turns this into a practical checklist and says which term each diagnostic checks.

12 Practical checklist and diagnostics

12.1 Checklist

1. **State the target** (§3.5) and the cause of $q \neq p$, and take the matching row of the decision table (§10.6). Before setting a KL penalty toward the sampler, work out what it penalizes (§10.5).

2. **Remove A first**, with SFC by default (Chapter 5). If you keep ratios (§5.4): weight first, then center; use log probabilities; compute masks from the ratio alone (§8.6).
3. **Check q and the support**: compute q with the actual temperature and filtering rules; with truncation, check whether the trainer has mass outside the support of q (§5.2).
4. **Check D**: is each response’s coefficient “a positive constant times R_i minus a quantity that does not involve this response” (§8.1)? If not, see the rows for D in the decision table.
5. **Reduce variance last** (Chapter 9): more alternative tokens or an exact head, a baseline that centers the signal, a critic if affordable.

12.2 Diagnostics

Logs with only the sampled token’s $q(a)$ and the trainer’s $\log p(a)$ do not give $q - p$ over the vocabulary, so the diagnostics that use full logits, a top- k head, or alternative tokens need these logged in advance. SFC guarantees a zero mean **in expectation** over the main and the alternative tokens together, not on **a batch already sampled**.

Term	Question to check	Data needed	How to read it	Limitations
A	Is the multiplier zero-mean under q ?	Top- k head or full logits, or independent alternative tokens	Average $\nabla_{\theta} z(a) - \hat{m}_q(s)$ over many tokens; zero within the standard error	Need not be zero on a single batch
A	Does shifting the baseline change the expected update?	Independent batches (main and alternative tokens resampled)	Add Δ to the signal (baseline and masks unchanged): no average change without A, Δ times the average $m_q - m_p$ for the plain estimator	One nonzero batch does not mean a bug
A	Bias left by a top- k head?	Head probabilities; a tail assumption or tail samples	$\bar{A}_q$ times the residual of Eq. (31)	Approximate centering is not exactly zero-mean
A	Do truncated weights bring A back?	Truncated weights; the trainer’s head probabilities	$\mathbb{E}_q[\tilde{w}g]$ in logit space should be near zero	In general not zero without re-centering
A	How large a share of the update is A?	Plain and SFC updates on one batch	Their difference $\sum_i A_i(\hat{m}_q - m_p)(s_i)$ estimates A; compare its norm with the SFC update	Average over several batches
A	Length-reward correlation in a group	Lengths and rewards	$\text{Cov}_G(L, R)$; the net push before and after length weighting	The coefficient of A only if the drift vectors are about the same everywhere (Appendix B.4)

Term	Question to check	Data needed	How to read it	Limitations
B, variance	Variance cost of the ratios	Logged ratios	Per-token effective sample size and D_2 : how concentrated the ratios are, not directly the variance factor (§9.4)	Truncation can hide a growing uncorrected part
B, C	Shares of the engine and the lag	Sampler–trainer KL at zero lag and at the normal lag	A large first value means the engine difference alone is not negligible	The KL is not a sum of the two parts
D	How large is the length bias?	The coefficient c_i as a function of L_i	Length elasticity $\partial \log c_i / \partial \log L_i$ or the implied per-token cost, against the explicit length penalty	Meaningful only at fixed R_i

13 Questions and short answers

About A

Q: Is A just a biased baseline? Why not choose the right baseline, or weight it by length?

Yes, but only together with a nonzero mean of the multiplier $\bar{f}$: A is the common offset $\bar{A}_q = V^q(s) - \mathbb{E}[b | s]$ times $\bar{f}$, and for plain REINFORCE $\bar{f}$ is nonzero only off-policy. Removing $\bar{A}_q$ at every prefix needs an accurate critic: a GRPO-style baseline has no offset at the prompt, but about $V^q(s_t) - V^q(s_0)$ in the middle of a response, however large the group is (§4.2). A length-weighted baseline makes only the net push of each group zero; the prefix-level part remains (§4.4).

Q: Is SFC just a better baseline, or a control variate?

No. A baseline adds a constant times the multiplier, which leaves the expectation unchanged only when the multiplier has mean zero; SFC changes the multiplier itself, so that its mean under the sampler is zero. It is a bias correction, after which the baseline is again a valid control variate (§5.1, §9.2; Marek and Ryabinin [1] say the same in their §4; Rush [6] introduces the baseline as a control variate).

Q: Can IS also remove A? Does “adding IS on top of SFC” mean multiplying the SFC score by the ratio?

Exact IS can: $\mathbb{E}_q[rg] = \mathbb{E}_p[g] = 0$ when the sampler covers the trainer’s support. A comes back with truncated or masked ratios and no re-centering, with a sampler that cannot draw tokens the trainer would choose, or when the score is centered before it is multiplied by the ratio (then with the opposite sign). The correct order is weight first, then center: with exact ratios the centering subtracts zero, so the correct combination is just exact IS (§5.4).

About B and C

Q: With SFC, is TIS still needed?

They fix different terms: SFC removes A; TIS fixes B and usually makes A smaller, but removing A requires re-centering. If the target is the deployed rollout engine, its error follows the fixed-error model, and the centering is accurate, SFC is the gradient of this target and B needs no fix; if the target is the current trainer, the answer depends on the lag and the effective sample size (§10).

About the guarantee

Q: Why can't the TRPO-style bound show that SFC is better?

The bound is a lower bound on the return of a given candidate policy, and for the same candidate the exact bounds for SFC and for the plain surrogate coincide; A, a property of the estimator, is not in the bound. What it does guarantee is that the new policy is better than the sampler whenever the gain in the surrogate exceeds the sum of the two penalties, without knowing the offset (§11).

A Derivations

A.1 The softmax score

For a softmax, $\log p(v) = z(v) - \log \sum_u e^{z(u)}$, whose second term has gradient $\sum_u p(u) \nabla_\theta z(u) = m_p$, so $g(v) = \nabla_\theta z(v) - m_p$. Its average is $m_p - m_p = 0$ under p and $m_q - m_p$ under q .

A.2 The score identity, the policy gradient, and random baselines

For any function $\phi(y)$ that does not depend on θ ,

$$\nabla_\theta \mathbb{E}_{y \sim p_\theta}[\phi(y)] = \sum_y \phi(y) \nabla_\theta p_\theta(y) = \sum_y \phi(y) p_\theta(y) \nabla_\theta \log p_\theta(y) = \mathbb{E}_{y \sim p_\theta}[\phi(y) S(y)],$$

where $S(y) = \nabla_\theta \log p_\theta(y \mid x) = \sum_t g(s_t, y_t)$ is the score of the whole response. Taking $\phi = R$ gives $\nabla_\theta J = \mathbb{E}_p[RS]$, which is REINFORCE. Conditioning each term of $RS = \sum_t R g(s_t, y_t)$ on $(s_t, y_t = v)$ replaces R with $\mathbb{E}[R \mid s_t, v] = Q^p(s_t, v)$, so

$$\nabla_\theta J = \sum_t \mathbb{E}_{s_t \sim p} \left[\sum_v p(v \mid s_t) Q^p(s_t, v) g(s_t, v) \right] = \sum_t \mathbb{E}_{s_t \sim p} \left[\sum_v p(v \mid s_t) A^p(s_t, v) g(s_t, v) \right],$$

where the last step uses $\sum_v p(v)g(v) = 0$ to remove $V^p(s_t)$. This is Eq. (5).

For a random baseline, $\mathbb{E}[(R_i - b_i)S_i] = \nabla_\theta J - \mathbb{E}[b_i S_i]$, which is Eq. (10), and $\mathbb{E}[b_i S_i] = \mathbb{E}[b_i] \mathbb{E}[S_i] = 0$ if b_i and y_i are independent. For one response's coefficient (Eq. (37)), fix the other responses: c_i is then a function of y_i , and $\phi = c_i$ gives $\mathbb{E}_{y_i} [c_i S_i] = \nabla_\theta \mathbb{E}_{y_i \sim p_\theta} [c_i]$; then average over them.

A.3 The gradient of the KL to the sampler

With q fixed, $D_{\text{KL}}(q \parallel p_\theta) = \sum_v q(v) [\log q(v) - \log p_\theta(v)]$, so $\nabla_\theta D_{\text{KL}}(q \parallel p_\theta) = -\sum_v q(v) g(v) = -(m_q - m_p)$.

A.4 Deriving the umbrella formula for variable-length responses

Write $\hat{g} = \sum_{t=1}^L (R - b) f_t$ as $\sum_{t \geq 1} \mathbf{1}\{L \geq t\} (R - b) f_t$. The event $\{L \geq t\}$ is determined by the prefix s_t , so the expectation of the t -th term is an average over the prefixes s_t that exist, written $\mathbb{E}_{s_t \sim q}$, with total mass $P_q(L \geq t)$. The five steps of §3.2 then hold word for word (in Step 2, $Q^q(s_t, v)$ already covers v being the end-of-sequence token), and the true gradient has the same form, $\sum_t \mathbb{E}_{s_t \sim p} [G_{s_t}^p[p]]$, so Eqs. (17) and (18) hold for any length.

A.5 The sampler's score under engine error

Let $q_\theta = \text{softmax}(z_\theta + \varepsilon)$. Appendix A.1 with the logits $z_\theta + \varepsilon$ gives $\nabla_\theta \log q_\theta(v) = \nabla_\theta z(v) + \nabla_\theta \varepsilon(v) - \sum_u q(u) (\nabla_\theta z(u) + \nabla_\theta \varepsilon(u))$, which regroups into Eq. (45). If $\nabla_\theta \varepsilon = 0$, the score is h .

A.6 A local family of distributions: the SFC score is a genuine score

Fix the trainer at θ_0 and, for a small parameter change δ , define

$$\tilde{q}_\delta(v) = \frac{1}{Z(\delta)} q(v) \frac{p_{\theta_0+\delta}(v)}{p_{\theta_0}(v)}, \quad Z(\delta) = \sum_u q(u) \frac{p_{\theta_0+\delta}(u)}{p_{\theta_0}(u)}.$$

Then $\tilde{q}_0 = q$, and differentiating the log at $\delta = 0$ gives $\nabla_\delta \log \tilde{q}_\delta(v)|_0 = g(v) - \mathbb{E}_q[g] = h(v)$, where the subtracted mean comes from differentiating Z . So h is the score of this family, and $\mathbb{E}_q[Ah]$ is on-policy REINFORCE on it. Because $\tilde{q}_\delta \propto \text{softmax}(z_{\theta_0+\delta} + z_q - z_{\theta_0})$, where z_q is the sampler's logits, this family is the same function as the straight-through (STE) construction, and under the fixed-error model it equals $q_{\theta_0+\delta}$ exactly.

A.7 The B term at the level of the objective

At one prefix, for a candidate policy π , write $r = \pi/q$ on the tokens with $q > 0$. The surrogate that IS ascends is $\mathbb{E}_\pi[A^q]$, and the one that SFC ascends is $\mathbb{E}_q[A^q \log r]$.

When the support is covered ($q > 0$ wherever $\pi > 0$), $\mathbb{E}_\pi[A^q] = \mathbb{E}_q[rA^q]$. Because $\mathbb{E}_q[A^q] = 0$,

$$\mathbb{E}_q[rA^q] - \mathbb{E}_q[A^q \log r] = \mathbb{E}_q[A^q (r - 1 - \log r)] = \mathbb{E}_q[A^q \phi(r)], \quad \phi(r) = r - 1 - \log r \geq 0,$$

and $\mathbb{E}_q[\phi(r)] = \mathbb{E}_q[r] - 1 - \mathbb{E}_q[\log r] = D_{\text{KL}}(q||\pi)$. Because $\phi \geq 0$ and $|A^q| \leq A_{\text{max}}$, the difference is at most $A_{\text{max}}D_{\text{KL}}(q||\pi)$ in absolute value.

When the support is not covered. Let π put mass ν on the tokens with $q = 0$. Then $\mathbb{E}_q[r] = 1 - \nu$ and $\mathbb{E}_\pi[A^q] = \mathbb{E}_q[rA^q] + \sum_{v: q(v)=0} \pi(v)A^q(v)$. The first part minus the SFC surrogate is still $\mathbb{E}_q[A^q \phi(r)]$, but now $\mathbb{E}_q[\phi(r)] = D_{\text{KL}}(q||\pi) - \nu$; the second part is at most $A_{\text{max}}\nu$ in absolute value. Adding the two, the difference is again at most $A_{\text{max}}D_{\text{KL}}(q||\pi)$ in absolute value. §6 shows when this bound is second order, with an example at the boundary of the support where it is not.

A.8 The B term at the level of the gradient

For a fixed prefix and signal, with $B_{\text{max}} = \max_v \|A^q(v)g(v)\|$ and Pinsker's TV $\leq \sqrt{D_{\text{KL}}/2}$,

$$\begin{aligned} \|G[q] - G[p]\| &= \left\| \sum_v (q(v) - p(v)) A^q(v) g(v) \right\| \leq B_{\text{max}} \sum_v |q(v) - p(v)| \\ &= 2B_{\text{max}} \text{TV}(q, p) \leq B_{\text{max}} \sqrt{2D_{\text{KL}}(q||p)}. \end{aligned}$$

A.9 Truncated and masked weights lose the zero mean

For TIS, $\tilde{w} = \min(r, \bar{r})$, $r - \tilde{w} = (r - \bar{r})_+$, and $\mathbb{E}_q[rg] = 0$, so $\mathbb{E}_q[\tilde{w}g] = -\mathbb{E}_q[(r - \tilde{w})g] = -\sum_{v: r(v) > \bar{r}} q(v)(r(v) - \bar{r})g(v)$, which is Eq. (32). For MIS, $\tilde{w} = r \mathbf{1}\{r_{\text{min}} \leq r \leq r_{\text{max}}\}$ and $\mathbb{E}_q[\tilde{w}g] = -\sum_{v \text{ masked}} p(v)g(v)$. Truncation does not always make A smaller than for plain REINFORCE: with $\bar{r} = 2$, $q = (0.25, 0.5, 0.25)$, $p = (0.25, 0.1, 0.65)$, and one parameter that moves only the logit of token 1, $\mathbb{E}_q[g] = 0$ (no A for plain REINFORCE here), but $\mathbb{E}_q[\tilde{w}g] = 0.0375$.

A.10 After weighting: logits or log-probabilities

With log-probabilities, the re-centered weighted loss has expected update $\mathbb{E}_q[A(\tilde{w}g - \mathbb{E}_q[\tilde{w}g])]$. Writing it with logits replaces g with $\nabla_\theta z = g + m_p$:

$$\mathbb{E}_q[A(\tilde{w}\nabla_\theta z - \mathbb{E}_q[\tilde{w}\nabla_\theta z])] = \mathbb{E}_q[A(\tilde{w}g - \mathbb{E}_q[\tilde{w}g])] + m_p(\mathbb{E}_q[A\tilde{w}] - \bar{A}_q \mathbb{E}_q[\tilde{w}]),$$

where the last factor in parentheses is $\text{Cov}_q(A, \tilde{w})$. With exact weights the second term equals $\mathbb{E}_p[A^q] m_p$ (Example 2: §5.4). Without weights ($\tilde{w} = 1$) it is zero, so plain SFC can use logits.

A.11 The conditional mean of the GRPO signal

The signal of the i -th response depends on the rewards of the whole group, $A_i = \mathcal{A}(R_i, R_{-i})$, where R_{-i} are the rewards of the other responses. With the group mean only, $\mathcal{A}(R_i, R_{-i}) = R_i - \frac{1}{n}(R_i + \sum_{j \neq i} R_j)$; with the standard deviation, this is further divided by the group's population standard deviation (dividing by n), and the signal is zero when all rewards in the group are equal.

Given that the i -th response sampled token v at prefix s , the other responses are independent rollouts of the same prompt, so

$$\mathbb{E}[A_i \mid s, v] = \mathbb{E}[\mathbb{E}[\mathcal{A}(R_i, R_{-i}) \mid R_i, s, v] \mid s, v] = \mathbb{E}_q[\chi(R_i) \mid s, v], \quad \chi(R_i) = \mathbb{E}_{R_{-i}}[\mathcal{A}(R_i, R_{-i})].$$

The multiplier depends only on (s, v) and on the independent alternative tokens, so in the expected update A_i can be replaced by $\mathbb{E}_q[\chi(R) \mid s, v]$, and $\mathbb{E}[\chi(R)] = 0$ (§4.2).

- Subtracting only the group mean: $R_i - \bar{R} = (1 - \frac{1}{n})(R_i - \bar{R}_{-i})$, so $\chi(R) = (1 - \frac{1}{n})(R - V^q(s_0))$.
- Rewards in $\{0, 1\}$: χ takes only the two values $\chi(0)$ and $\chi(1)$, so $\chi(R) = \chi(0) + \zeta R$ with $\zeta = \chi(1) - \chi(0) > 0$, and $\mathbb{E}[\chi(R)] = 0$ gives $\chi(R) = \zeta(R - V^q(s_0))$. For $n = 2, 4, 8$, $\zeta = 1.00, 1.62, 1.86$ at $V^q(s_0) = 0.5$, and $1.00, 1.66, 2.11$ at $V^q(s_0) = 0.2$ or 0.8 .
- Rewards with three or more values: χ is in general not linear. With $n = 2$ and rewards independent and uniform on $\{0, 1, 10\}$, dividing by the standard deviation gives $A_i = \text{sign}(R_i - R_j)$, so $(\chi(0), \chi(1), \chi(10)) = (-2/3, 0, 2/3)$, while $R - \mathbb{E}[R]$ at these values is $(-11/3, -8/3, 19/3)$.

A.12 The length-weighted baseline: a few identities

The identity for a shared baseline (Eq. (39)). For every i , $\mathbb{E}[R_i S_i] = \nabla_\theta J$. The sum $\sum_i S_i = \nabla_\theta \log \prod_i p_\theta(y_i \mid x)$ is the score of the whole group, so by the score identity $\mathbb{E}[\bar{b} \sum_i S_i] = \nabla_\theta \mathbb{E}[\bar{b}]$. Subtracting gives the identity.

The length-weighted mean is the ordinary mean plus a covariance term. Since $\frac{1}{n} \sum_i L_i R_i = \bar{L} \bar{R} + \text{Cov}_G(L, R)$, where $\text{Cov}_G(L, R) = \frac{1}{n} \sum_i (L_i - \bar{L})(R_i - \bar{R})$, dividing by $\bar{L}$ gives $b_{\text{LW}} = \bar{R} + \text{Cov}_G(L, R)/\bar{L}$. Substituting into Eq. (39) gives Eq. (40).

The covariance under 0/1 rewards (§8.3). $\mathbb{E}[L] = \varrho L_c + (1 - \varrho)L_w$ and $\mathbb{E}[LR] = \varrho L_c$, so $\text{Cov}(L, R) = \varrho L_c - \varrho \mathbb{E}[L] = \varrho(1 - \varrho)(L_c - L_w)$; dividing by $\mathbb{E}[L]$ gives the formula there.

The length-weighted mean is the least-squares constant over tokens. Let R_{it} be the reward of the t -th token of the i -th response. Minimizing $\sum_{i,t} (R_{it} - b)^2$ over b gives $b = \sum_{i,t} R_{it} / \sum_i L_i$; when the reward is not assigned per token, $R_{it} = R_i$, and this is b_{LW} .

The net push left in the leave-one-out version. $R_i - b_{-i}^{\text{LW}} = (R_i - b_{\text{LW}})/(1 - \omega_i)$, and subtracting $\sum_i L_i(R_i - b_{\text{LW}}) = 0$ gives $\sum_i L_i(R_i - b_{-i}^{\text{LW}}) = \sum_i L_i A_i^{\text{LW}} \frac{\omega_i}{1 - \omega_i}$, where $A_i^{\text{LW}} = R_i - b_{\text{LW}}$; §5.6 gives its size.

A.13 The expected update need not be the gradient of a fixed objective

If the expected update $G(\theta)$ were the gradient of a fixed scalar function $\Phi(\theta)$, the Jacobian of G would be the Hessian of Φ and so symmetric. Take groups of two responses, three possible responses with rewards $R = (0, 1, 2)$ and lengths $L = (1, 2, 4)$, and $p_\theta = \text{softmax}(\theta_1, \theta_2, 0)$. With DAPO-style normalization, $c_i = (R_i - \bar{R})/(L_i + L_j)$ with $\bar{R} = (R_i + R_j)/2$, the expected update is $G(\theta) = \mathbb{E}[\sum_i c_i S_i]$. At $\theta = 0$, $\partial G_1/\partial \theta_2 - \partial G_2/\partial \theta_1 = -1/135 \neq 0$, so there is no Φ with $G = \nabla_\theta \Phi$. We computed this value exactly with rational arithmetic and checked it with finite differences.

A.14 The variance cost V_K

As in §9.4, $\hat{X} = X_0 - Ae$, and (i) $\mathbb{E}[e] = 0$; (ii) e is independent of the main token and its signal; (iii) $\text{Cov}(e) = \Sigma_g/K$. Then $\mathbb{E}[\hat{X}] = \mathbb{E}[X_0] - \mathbb{E}[A]\mathbb{E}[e] = \mathbb{E}[X_0]$. In $\text{tr Cov}(X_0 - Ae) = \text{tr Cov}(X_0) + \text{tr Cov}(Ae) - 2\mathbb{E}[(X_0 - \mathbb{E}X_0)^\top Ae]$, the cross term equals $\mathbb{E}[(X_0 - \mathbb{E}X_0)A]^\top \mathbb{E}[e] = 0$ by (ii) and (i), and as $\mathbb{E}[Ae] = 0$, $\text{tr Cov}(Ae) = \mathbb{E}[A^2\|e\|^2] = \mathbb{E}[A^2]\mathbb{E}\|e\|^2 = \mathbb{E}[A^2]\text{tr } \Sigma_g/K$. So $\text{tr Cov}(\hat{X}) = \text{tr Cov}(X_0) + V_K$.

A.15 The covariance between leave-one-out updates in the same group

On-policy, take a group of n independent responses with $X_i = (R_i - b_{-i})S_i$, $b_{-i} = \frac{1}{n-1} \sum_{k \neq i} R_k$, and $m = \nabla_\theta J = \mathbb{E}[R_i S_i]$. For $i \neq j$, expand $(R_i - b_{-i})(R_j - b_{-j}) = R_i R_j - R_i b_{-j} - b_{-i} R_j + b_{-i} b_{-j}$ and multiply by $S_i S_j^\top$. A part in which S_j appears without R_j (or S_i without R_i) has zero mean, because the responses are independent and $\mathbb{E}[S_j] = 0$. This removes the second term (b_{-j} does not contain R_j) and the third, and leaves only $R_j R_i/(n-1)^2$ of the fourth, so

$$\mathbb{E}[X_i X_j^\top] = \left(1 + \frac{1}{(n-1)^2}\right) m m^\top, \quad \text{Cov}(X_i, X_j) = \frac{m m^\top}{(n-1)^2},$$

and summing over the pairs gives the variance of the group average in §9.1.

A.16 A numerical check of the sequence-level ratio

Step 1 chooses route X or Y: $q_1 = (0.9, 0.1)$ and $p_1 = (0.5, 0.5)$. Step 2 chooses one of two tokens on each route, with $q_2(\cdot | X) = (0.7, 0.3)$, $q_2(\cdot | Y) = (0.4, 0.6)$, $p_2(\cdot | X) = (0.5, 0.5)$, and $p_2(\cdot | Y) = (0.6, 0.4)$. The rewards are $R(X, \cdot) = (1.0, 0.2)$ and $R(Y, \cdot) = (0.3, 0.9)$, and the parameters are the log-odds θ_1 of route X and θ_2^X, θ_2^Y of the first token on each route.

The true gradient is $\partial J/\partial\theta_1 = 0.015$ and $\partial J/\partial\theta_2 = (0.100, -0.072)$. On the sampler’s data, the whole-response ratio $p(y)/q(y)$ gives all three exactly; the ratio of each token’s own step gives 0.025 for θ_1 (error from the continuation, C2) and $(0.180, -0.014)$ for θ_2 (error from the prefix distribution, C1: route X is 90% of the data but half under p).

A.17 The first-order effect of drift on entropy

When the parameters are the logits, $\mathcal{H} = -\sum_u p(u) \log p(u)$ has derivative $\partial\mathcal{H}/\partial z(v) = -p(v)(\log p(v) + \mathcal{H})$. Multiplying by the change $\delta z = \lambda \bar{A}_q(q - p)$ that A makes to the logits, summing over v , and using $\mathcal{H} = -\mathbb{E}_p[\log p]$ gives $\Delta\mathcal{H} \approx \lambda \bar{A}_q \text{Cov}_p(\log p, p - q)$. With two tokens, any functions X and Y have $\text{Cov}_p(X, Y) = p_1(1 - p_1)(X_1 - X_2)(Y_1 - Y_2)$, and $Y = p - q$ has $Y_1 - Y_2 = 2(p_1 - q_1)$, which gives the formula of Appendix B.3.

A.18 Stability of the feedback-loop model

Substituting $\theta(t) = vt + \delta(t)$ into the model of Appendix B.2 gives the constant speed $v = \lambda\Psi/(1+\Gamma)$, with $\Gamma = \lambda\tau\bar{A}_qF \neq -1$, and the deviation equation $\dot{\delta}(t) = -k(\delta(t) - \delta(t - \tau))$ with $k = \lambda\bar{A}_qF$. Trying $\delta = e^{\varsigma t}$ (ς may be complex) gives $\varsigma = -k(1 - e^{-\varsigma\tau})$. The root $\varsigma = 0$ only shifts θ by a constant; the other roots decide stability:

- $k > 0$ ($\bar{A}_q > 0$): in $\varsigma + k = ke^{-\varsigma\tau}$ with $\text{Re } \varsigma \geq 0$ and $\varsigma \neq 0$, the left side has modulus $|\varsigma + k| > k$ and the right side at most k . So no root is unstable, and the speed converges to v .
- $k < 0$ ($\bar{A}_q < 0$): for $\varsigma \neq 0$ the equation becomes $1 = |k| \int_0^\tau e^{-\varsigma u} du$, whose right side has modulus at most $|k|\tau = |\Gamma|$ when $\text{Re } \varsigma \geq 0$. So for $|\Gamma| < 1$ the speed converges to v , faster than without A. For $|\Gamma| > 1$, the function $|k|(1 - e^{-\varsigma\tau}) - \varsigma$ of real ς is zero at $\varsigma = 0$, has derivative $|\Gamma| - 1 > 0$ there, and tends to $-\infty$ as $\varsigma \rightarrow \infty$, so it has a positive root ς^* , and the deviation grows like $e^{\varsigma^* t}$. For $|\Gamma| = 1$ there is no constant-speed solution, $\varsigma = 0$ is a double root, and θ grows like t^2 .

B Optional reading: which training mechanisms may amplify term A

These four models [model] show what can happen, not rules that every training run obeys.

B.1 Adding up along the response

Assumptions. The reward is determined by the choices at a few “branch points”, and once the outcome is settled, later tokens no longer affect the return; $m_q(s_t) - m_p(s_t)$ is approximately the same vector u at every position. **Formula.** The useful signal appears only at the branch points ($A^q \approx 0$ elsewhere), while A appears on every token after the first branch point: over one response it is about $u \sum_t \bar{A}_q(s_t)$, summed over these tokens. **What follows.** The total useful signal does

not grow with length, while the total A grows roughly linearly with the number of tokens after the branch point, because its direction is similar on all of these tokens.

B.2 A feedback loop under staleness

Assumptions. The sampler is the trainer from τ steps ago; $m_q - m_p \approx -F(\theta_p - \theta_q)$, where $F = \text{Cov}_p(\nabla_{\theta} z)$ is the Fisher matrix (a first-order expansion); $\bar{A}_q$ and F stay fixed; and we look at one direction, so F is a scalar. λ is the learning rate and Ψ (a scalar) is the useful part of the update. **Formula.** The faster the trainer moves, the larger $m_q - m_p$ becomes, and A in turn changes the speed of the trainer:

$$\dot{\theta}(t) = \lambda[\Psi - \bar{A}_q F(\theta(t) - \theta(t - \tau))], \quad \text{constant-speed solution} \quad \dot{\theta} = \frac{\lambda\Psi}{1 + \Gamma}, \quad \Gamma = \lambda\tau\bar{A}_q F.$$

What follows (Appendix A.18). For $\Gamma > 0$ (that is, $\bar{A}_q > 0$) the loop is stable and the speed drops to $1/(1 + \Gamma)$ of its original value; for $-1 < \Gamma < 0$ it is stable and the speed increases; for $\Gamma = -1$ the speed grows linearly; for $\Gamma < -1$ the deviation grows exponentially. SFC makes $\Gamma = 0$, and a smaller learning rate, lag, or $|\bar{A}_q|$ (a better baseline) makes $|\Gamma|$ smaller (KL penalty: §10.5).

B.3 Coupling with entropy

Assumptions. The parameters are the logits, or close to them, so one step of A in logit space is $\lambda\bar{A}_q(q - p)$. **Formula.** This step changes the trainer’s entropy by $\Delta\mathcal{H} = \lambda\bar{A}_q \text{Cov}_p(\log p, p - q) + O(\lambda^2)$ (Appendix A.17). **What follows.** The sign is set by $\bar{A}_q$ and this covariance together. With two tokens, $p = (p_1, 1 - p_1)$ and $q = (q_1, 1 - q_1)$, the covariance $\text{Cov}_p(\log p, p - q) = 2p_1(1 - p_1)(p_1 - q_1) \log \frac{p_1}{1 - p_1}$ is positive when $p_1 > 1/2$ and $p_1 > q_1$; then $\bar{A}_q < 0$ lowers the entropy, and $\bar{A}_q > 0$ raises it.

B.4 The within-group coefficient can have the bad sign

Assumptions. We use a token-level loss, and within one group $m_q(s) - m_p(s)$ is approximately the same vector u at every position. **Formula.** The A of this group is about $u \sum_i L_i A_i = u \cdot n \text{Cov}_G(L, R)$ (Eq. (26)); averaged over tokens, the coefficient is $\bar{A}_{\text{tok}} = \text{Cov}_G(L, R)/\bar{L}$. **What follows.** With a length limit or a length penalty, long responses often get low reward, so $\text{Cov}_G(L, R) < 0$ and $\bar{A}_{\text{tok}} < 0$: this is the sign for which the feedback loop above can diverge.

C Implementation variants, and where other work fits in the umbrella formula

The implementations differ in how much of q is logged with each token; going down the table, the remaining A gets roughly smaller (an empirical ordering, not a guarantee):

What is logged	Method	Remaining A
Nothing	Plain REINFORCE	$\bar{A}_q(m_q - m_p)$
q on the top- k head	Official SFC code (default $k = 128$) [2]; verl (§C.1)	$\bar{A}_q q(\bar{H})(\mathbb{E}_{q(\cdot \bar{H})}[\nabla_\theta z] - \mathbb{E}_{p(\cdot \bar{H})}[\nabla_\theta z])$, from the tail only
K alternative tokens	The loss in §5.1	Zero in expectation; variance V_K
The full distribution	Exact centering	Zero

C.1 Logged head, modeled tail

Marek and Ryabinin [1, 2] log the sampler’s probabilities on the top- k set H and center with the $\hat{q}$ of Eq. (30), which in terms of the tail ratio ρ is Eq. (7) in their Appendix A.1:

$$\hat{q}(v) = \begin{cases} q(v), & v \in H, \\ \rho p(v), & v \in \bar{H}, \end{cases} \quad \rho = \frac{q(\bar{H})}{p(\bar{H})} = \frac{1 - \sum_{v \in H} q(v)}{1 - \sum_{v \in H} p(v)}.$$

Centering with $\hat{q}$ leaves the residual A of the table [identity]: $\bar{A}_q$ times the quantity of Eq. (31) (g and $\nabla_\theta z$ differ by m_p , the same in both conditional expectations), a fixed bias that depends on the baseline. With $k = 128$ the head covers more than 99.9% of the sampler’s probability on average, except for a 30B model with INT8 weights and activations and an INT4 KV cache (average 99.45%, worst batch 95.8%; their Appendix A.4). Sampling the tail is unbiased but adds variance (§9.4).

Unweighted. Since $\sum_v p(v)g(v) = 0$ gives $\sum_{v \in \bar{H}} \rho p(v)g(v) = -\rho \sum_{v \in H} p(v)g(v)$, the tail part of the mean score can be computed from the head:

$$\mathbb{E}_{\hat{q}}[g] = \sum_{v \in H} (q(v) - \rho p(v)) g(v), \quad \mathcal{L} = -\text{sg}(A) \left[\log p(a) - \sum_{v \in H} \text{sg}(q(v) - \rho p(v)) \log p(v) \right].$$

These are their Eqs. (8) and (6); backpropagation uses only the sampled token and the head.

Weighted. For a weighting rule $\tilde{w} = w(r)$ (for TIS, $w(r) = \min(r, \bar{r})$), the modeled tail has $p/\hat{q} = 1/\rho$, so its weight is $w(1/\rho)$. With $\psi = \rho w(1/\rho)$,

$$\mathbb{E}_{\hat{q}}[\tilde{w}g] = \sum_{v \in H} (q(v)\tilde{w}(v) - \psi p(v)) g(v),$$

$$\mathcal{L} = -\text{sg}(A) \left[\text{sg}(\tilde{w}(a)) \log p(a) - \sum_{v \in H} \text{sg}(q(v)\tilde{w}(v) - \psi p(v)) \log p(v) \right].$$

These are their Eqs. (10) and (12) (Appendices A.2–A.3; their $\alpha = \rho f(1/\rho)$ is our ψ) and the loss in the official README, with $w(r) = \min(r, 2)$ for TIS and $w(r) = r \mathbf{1}\{0.5 \leq r \leq 5\}$ for MIS [2]; the QingKe article [5] (§1.5) also derives them and describes the verl implementation.

The main token a gets $\tilde{w}(a) = w(p(a)/q(a))$ from its true logged $q(a)$, even in the tail (the official code keeps its log-probability for this); the unlogged tail tokens in the centering term use $w(p(v)/\hat{q}(v)) = w(1/\rho)$. The four rules give: unweighted, $\psi = \rho$; exact IS, $\psi = 1$; TIS, $\psi = \min(1, \bar{r}\rho)$; MIS, $\psi = \mathbf{1}\{r_{\min} \leq 1/\rho \leq r_{\max}\}$. With exact IS and $\rho > 0$ the subtracted term is zero; this fails if the sampler itself truncates to H ($\rho = 0$), where the tail ratio is undefined.

C.2 Where other work fits in the umbrella formula

Result	Where in this note	Source
A = distillation toward the sampler; SFC; top- k implementation; quantization and staleness experiments; SFC with TIS/MIS	§4.1; §5.1; §C.1; §10.1; §9.6	Marek and Ryabinin [1, 2]
SFC plus a KL to the sampler; Pinsker bound; coefficient decaying with age	§10.5; Appendix A.8	Zhang’s blog series [3]
A q -centered score times p/q loses its zero mean, so use KL, not IS	Agrees with §5.4	Zhang’s second post [3]
Sampler–learner KL (StableReinforce, default coefficient 0.01)	§10.5	Thinking Machines’ Tinker [4]
TRPO-style bound	Chapter 11: constrains B, C, not A	TRPO [22]
GRPO’s group mean and standard deviation	§4.2, §8.5, Appendix A.11	DeepSeekMath [10]
Length bias of $1/L_i$; difficulty bias; constant divisor	§8.2, §8.5	Dr. GRPO [12]
Token-level loss; clip-higher	§8.2; §8.6	DAPO [11]
PPO’s clipping	§8.6: not baseline-invariant	PPO [23]
Geometric mean of the sequence-level ratio	§7.3: outside Premise P; the gradient carries $1/L_i$	GSPO [13]
Leave-one-out baseline	§2.2	RLOO [19, 20]
Variance-optimal constant baseline; length-weighted baseline	§2.4; §5.6, §8.2	[17, 18]; OPO [14]
Baseline as a control variate	§9.2, Chapter 13	Rush [6]
TIS; MIS	§5.4	[7]; [8, 9]
The QingKe community article on five interpretations of score centering [5]		
Notation: its s ; its $m_q = \mathbb{E}_q[s]$; its B	Our g ; our $m_q - m_p$; our A	
Drift; its baseline-invariance section	The A term; §3.2	
Ideal baseline ($\bar{A}_q = 0$)	Plain REINFORCE equals SFC	
STE	The fixed-error model (§10.1, Appendix A.5)	The Zhihu article cited in its §4
Local family of distributions	Appendix A.6	
SC+TIS/MIS; top- k formula with the coefficient; verl implementation	Weight first, then center (§5.4); §C.1	

STE and the local family get the centering term by differentiating the normalizer of a q -anchored distribution; drift ($\bar{f} = 0$) and the ideal baseline ($\bar{A}_q = 0$) zero the two factors of $\bar{A}_q \cdot \bar{f}$.

D Notation

Symbol	Meaning	First used
x, y, L, s_t	prompt; response; its length; prefix before token t	§1.1
$q, p = p_\theta$	sampler (generated the data); trainer (being updated)	§1.1
$J(\theta), \nabla_\theta J$	expected return of the trainer; true gradient	§1.2
$z(s, v), \nabla_\theta z, m_\pi(s)$	logit of token v ; its gradient; mean of $\nabla_\theta z$ under π	§1.2
$g(s, v)$	trainer’s score $\nabla_\theta \log p(v s) = \nabla_\theta z - m_p$	§1.2
Q^π, V^π, A^π	action value, prefix value, true advantage under $\pi \in \{q, p\}$	§1.2
$G_s[\mu], G_s^p[\mu]$	expected update $\sum_v \mu(v) A^q(s, v) g(s, v)$; the same with A^p for A^q	§1.2
R, b, A	return; baseline; signal $R - b$ of one sample	§1.4
f, w, c	multiplier $wg - c$; token weight; centering vector	§1.4
ξ, f_ξ, f_0	extra randomness; multiplier $w_\xi g - c_\xi$ of a draw; its mean over ξ	§1.4
$r, \tilde{w}, \bar{r}$	ratio p/q ; truncated or masked ratio; ratio cap	§1.4
$\mu, \bar{f}, \bar{A}_q$	token weight in the update, $q \mathbb{E}_\xi[w_\xi]$; mean of the multiplier, $\mathbb{E}_q[f_0 s]$; common offset, $V^q(s) - \mathbb{E}[b s]$	§1.4
n, L_i, s_{it}	group size; length of response i ; its prefix before token t	§1
S_i	sum of the scores of response i , $\nabla_\theta \log p(y_i x)$	§2.2
b_{-i}, ω_i	leave-one-out mean; weight of R_i in the baseline of response i	§2.2
N, N^*	number of independent groups; N at which noise equals signal	§2.4
b_{var}	scalar baseline that minimizes the variance	§2.4
A, B, C, D	drift; token weights; occupancy; term beyond Premise P	§3.3
$\hat{g}_0$	reference estimator for D (violating part replaced to satisfy P)	§3.3
$C1, C2, L_q(\pi)$	prefix distribution; continuation (A^p for A^q); TRPO surrogate	§3.3
ε	logit error of the inference engine	§3.5
$\sigma^2(s, v)$	return variance from the continuation after token v	§3.8
Δ	constant added to all signals	§4.1
$\chi(R), \zeta$	mean of the GRPO signal given R ; its slope for binary rewards	§4.2
$\bar{L}, \text{Cov}_G, b_{\text{LW}}$	group mean length; group covariance; length-weighted mean	§4.4
$h(s, v), K, a'_k$	SFC score $\nabla_\theta z - m_q$; number of alternative tokens; the tokens	§5.1
$H, \bar{H}, \hat{q}$	logged head (such as the top- k); tail; approximate q of Eq. (30)	§5.1
$r_{\min}, r_{\max}$	ratio interval kept by MIS	§5.4
$B_{\max}, A_{\max}, \nu$	$\max_v \ A^q(v)g(v)\ $; $\max_v A^q(v) $; mass of π on tokens with $q = 0$	§6
$\bar{w}$	total mass of the truncated weights, $\sum_v \min(p, \bar{r}q) = \mathbb{E}_q[\bar{w}]$	§6.3
D_2	Rényi divergence of order 2	§7.2
$\gamma, d_\pi, \eta(\pi)$	discount; prefix distribution under π ; expected return of π	§7
c_i, ℓ_i	coefficient of response i ; its length factor	§8
$\theta_0, \mathcal{F}(\theta; \theta_0)$	current parameters; local surrogate of this update	§8.1
b_{-i}^{LW}	leave-one-out length-weighted mean	§8.2
ϱ, L_c, L_w	success rate; mean lengths of correct and of wrong responses	§8.3
$\bar{b}$	baseline shared by the whole group	§8.3
X_i	update from response i	§9.1
ESS	effective sample size $N/\mathbb{E}_q[r^2]$ (how concentrated the ratios are)	§9.4

Symbol	Meaning	First used
V_K, e, Σ_g	cost and error of estimating m_q ; covariance of $\nabla_{\theta} z$ under q	§9.4
κ	size of the lag in the toy	§9.6
β	KL penalty coefficient	§10.5
TV	total variation distance	§11
F, λ, τ	Fisher matrix; learning rate; lag in steps	§B.2
Γ, Ψ	feedback-loop gain; useful part of the update	§B.2
$\mathcal{H}$	entropy of the trainer	§B.3

References

- [1] M. Marek and M. Ryabinin. Score Centering Stabilizes Off-policy Reinforcement Learning. arXiv:2609.20807, 2026. <https://arxiv.org/abs/2609.20807> (equation numbers cited in this note follow the arXiv PDF).
- [2] M. Marek. score-centering: official code for the paper above. We checked `README.md` and `losses.py` at commit 6484296 (October 15, 2026). <https://github.com/martin-marek/score-centering/blob/64842967077ea105470ca73fc5d714366d3f7bdb/losses.py>
- [3] Y. Zhang (Yihua Zhang). A series of three blog posts on score function centering, October 2026: (1) Score Function Centering Made Simple: Understanding Trainer-Generator Mismatch in Async RL from the Score Function; (2) Score Function Centering: Why a Stale Rollout Does Not Have to Be Importance Sampled; (3) Stabilize Your Async RL Training with Score Function Centering and Trainer-Generator KL. <https://normaluhr.github.io/en/p/score-function-centering/> (also on Hugging Face); code and experiment logs: <https://github.com/NormalUhr/score-function-centering-trainer-generator-kl>
- [4] Thinking Machines Lab. StableReinforce (`stable_reinforce`), Tinker documentation, alpha, 2026. <https://tinker-docs.thinkingmachines.ai/tinker/losses/stable-reinforce/>
- [5] Palind. 关于 Score Centering 的五种解释: drift、BPO、理想 baseline、STE、局部分布族 (Five interpretations of score centering: drift, BPO, the ideal baseline, STE, and a local family of distributions). In Chinese. QingKe community (青稞社区), October 13, 2026. <https://qingkeai.online/blog/five-Score-Centering>
- [6] S. Rush. From Sampling to Reinforce. Blog post, 2026. <https://srush.github.io/sampling-to-reinforce/>
- [7] F. Yao, L. Liu, D. Zhang, C. Dong, J. Shang, and J. Gao. Your Efficient RL Framework Secretly Brings You Off-Policy RL Training. Blog post, August 2025, <https://fengyao.notion.site/off-policy-rl>. The corresponding paper is On the Rollout-Training Mismatch in Modern RL Systems, NeurIPS 2025 Workshop on Efficient Reasoning.
- [8] Ling Team. Every Step Evolves: Scaling Reinforcement Learning for Trillion-Scale Thinking Model. arXiv:2510.18855, 2025. (IcePop, §2.3.2: tokens whose ratio falls outside an interval get zero weight.)
- [9] J. Liu, Y. Li, et al. When Speed Kills Stability: Demystifying RL Collapse from the Training-Inference Mismatch. Blog post, September 2025, <https://richardli.xyz/rl-collapse> (sequence-level masked IS appears in the follow-up Part 3).
- [10] Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, X. Bi, H. Zhang, M. Zhang, Y. K. Li, Y. Wu, and D. Guo. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. arXiv:2402.03300, 2024.
- [11] Q. Yu, Z. Zhang, R. Zhu, et al. DAPO: An Open-Source LLM Reinforcement Learning System at Scale. arXiv:2503.14476, 2025.
- [12] Z. Liu, C. Chen, W. Li, P. Qi, T. Pang, C. Du, W. S. Lee, and M. Lin. Understanding R1-Zero-Like Training: A Critical Perspective. In *COLM*, 2025. arXiv:2503.20783.
- [13] C. Zheng, S. Liu, M. Li, X.-H. Chen, B. Yu, C. Gao, K. Dang, Y. Liu, R. Men, A. Yang, J. Zhou, and J. Lin. Group Sequence Policy Optimization. arXiv:2507.18071, 2025.
- [14] Y. Hao, L. Dong, X. Wu, S. Huang, Z. Chi, and F. Wei. On-Policy RL with Optimal Reward Baseline. arXiv:2505.23585, 2025.
- [15] R. J. Williams. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Machine Learning*, 8(3-4):229–256, 1992.
- [16] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *Advances in Neural Information Processing Systems 12 (NIPS 1999)*, 2000.

- [17] L. Weaver and N. Tao. The Optimal Reward Baseline for Gradient-Based Reinforcement Learning. In *UAI*, pages 538–545, 2001.
- [18] E. Greensmith, P. L. Bartlett, and J. Baxter. Variance Reduction Techniques for Gradient Estimates in Reinforcement Learning. *Journal of Machine Learning Research*, 5:1471–1530, 2004.
- [19] W. Kool, H. van Hoof, and M. Welling. Buy 4 REINFORCE Samples, Get a Baseline for Free! ICLR 2019 Workshop on Deep RL Meets Structured Prediction, 2019.
- [20] A. Ahmadian, C. Cremer, M. Gallé, M. Fadaee, J. Kreutzer, O. Pietquin, A. Üstün, and S. Hooker. Back to Basics: Revisiting REINFORCE-Style Optimization for Learning from Human Feedback in LLMs. In *ACL*, pages 12248–12267, 2024.
- [21] S. M. Kakade and J. Langford. Approximately Optimal Approximate Reinforcement Learning. In *ICML*, pages 267–274, 2002.
- [22] J. Schulman, S. Levine, P. Moritz, M. Jordan, and P. Abbeel. Trust Region Policy Optimization. In *ICML*, PMLR 37:1889–1897, 2015.
- [23] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal Policy Optimization Algorithms. arXiv:1707.06347, 2017.