Weight-Preserving Simulated Tempering

Jeffrey S. Rosenthal, University of Toronto.

(Fujitsu/UofT/DA Monthly Seminar, March 23, 2022)

N.G. Tawn, G.O. Roberts, and J.S. Rosenthal, "Weight-Preserving Simulated Tempering". *Statistics and Computing* **30** (2020), 27–41.

G.O. Roberts, J.S. Rosenthal, and N.G. Tawn, "Skew Brownian Motion and Complexity of the ALPS Algorithm". *Journal of Applied Probability* **59(3)**, to appear.

Background on the Metropolis Algorithm (MCMC)

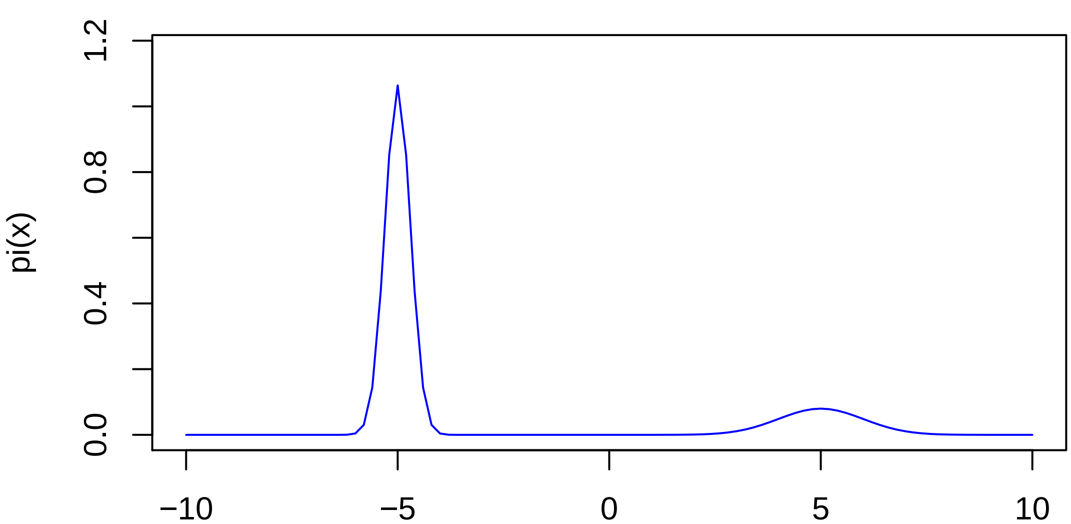
- Given a previous state X, propose a new state $Y \sim Q(X, \cdot)$.

(Assume that Q is symmetric about X; otherwise "Metropolis-Hastings".)

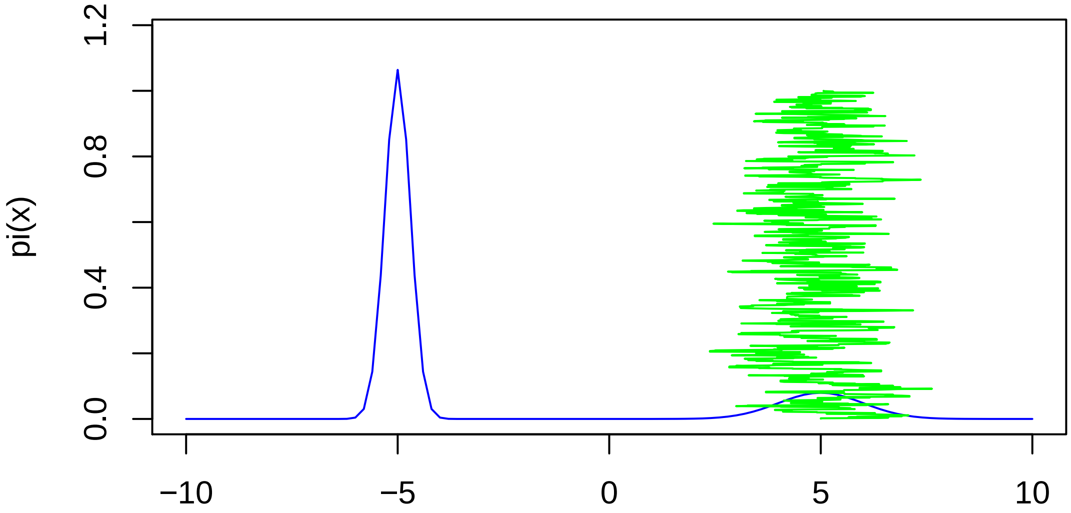
- Then, if $\pi(Y) > \pi(X)$, accept the new state and move to it.
- If not, then accept it only with probability $\pi(Y) / \pi(X)$, otherwise reject it and stay where you are.
- The empirical distribution (black) converges to the target (blue). [Metropolis]
- Good for sampling (to estimate expected values $\mathbf{E}_\pi(h)$), and for optimisation (to find modes $\arg\max_x \pi(x)$).

Problem: The Chain can get Stuck in a Local Mode

- Can't "jump over" places where π small. [Metropolis ex]
- Consider the following running example, with two separated modes:



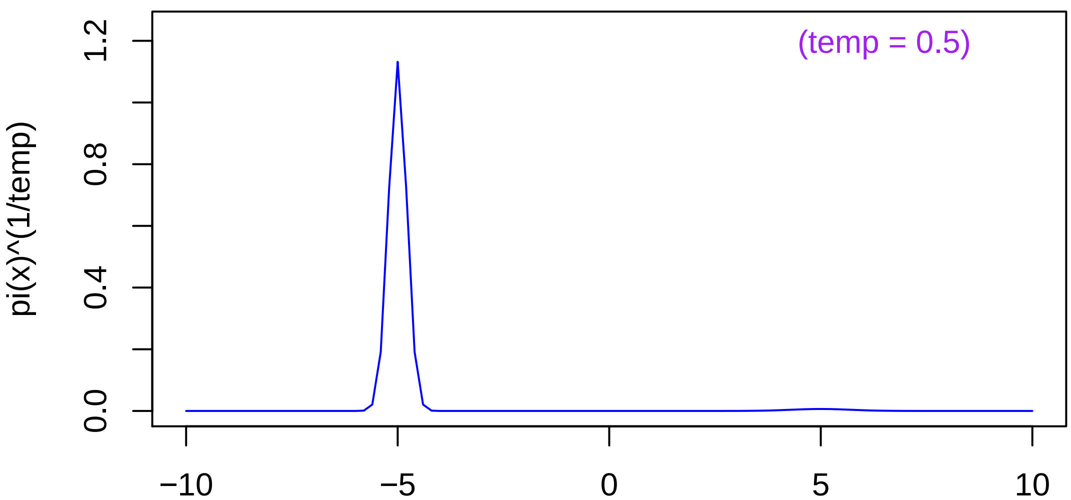
- A simple Metropolis algorithm may have trouble mixing well:



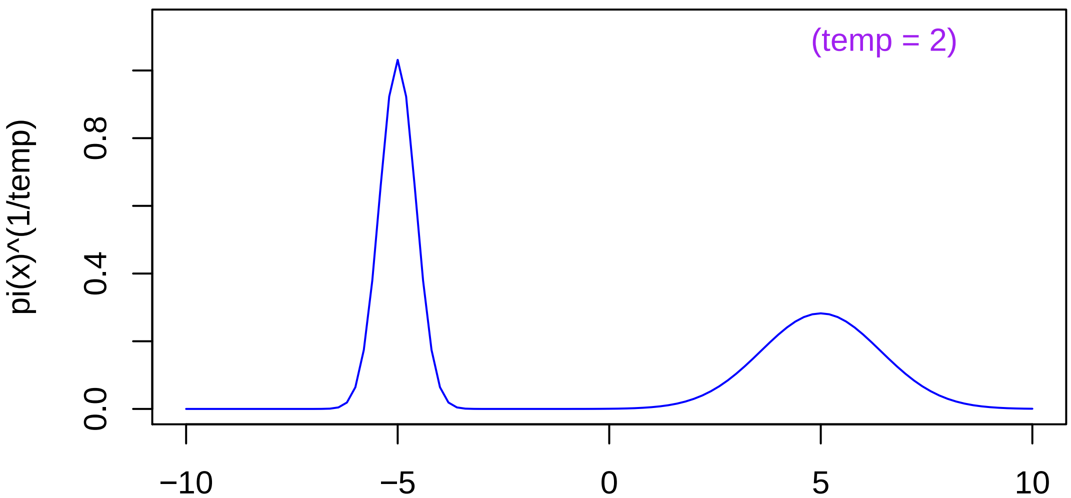
- The chain (green, running "up") can't easily move from "5" to "-5".
- And this problem gets even <u>worse</u> in higher dimensions.

Traditional Solution: Tempering

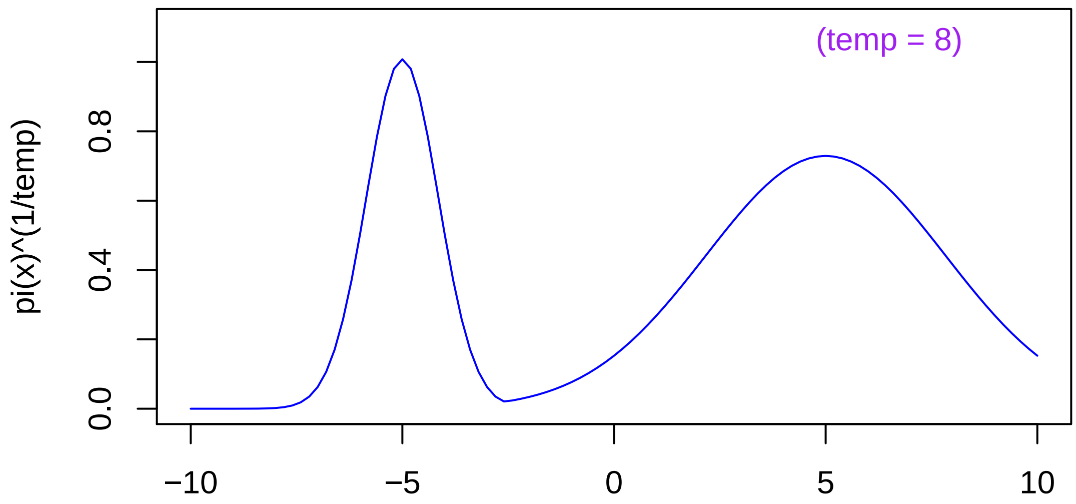
- Replace the target $\pi(x)$ by a <u>tempered</u> version, $\pi_\tau(x) = \pi(x)^{1/\tau}$.
- For optimisation: let $\tau \searrow 0$ (cooling), to make it more "peaked":



- But for mixing, take $\tau \gg 1$, to make it "flatter" $(\pi(x)^{1/\tau} \to 1)$:

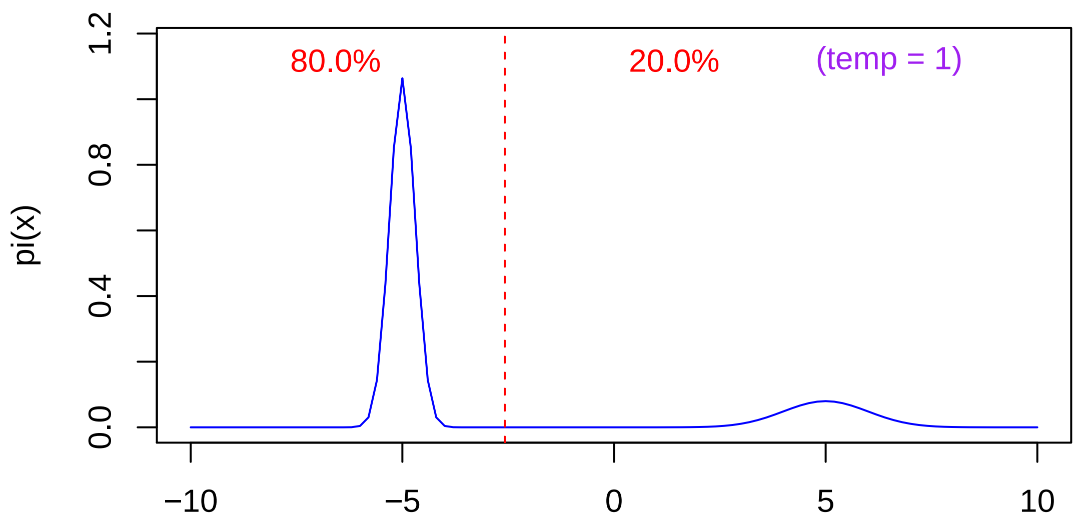


- If τ is large enough, then the chain can explore, without obstacles:

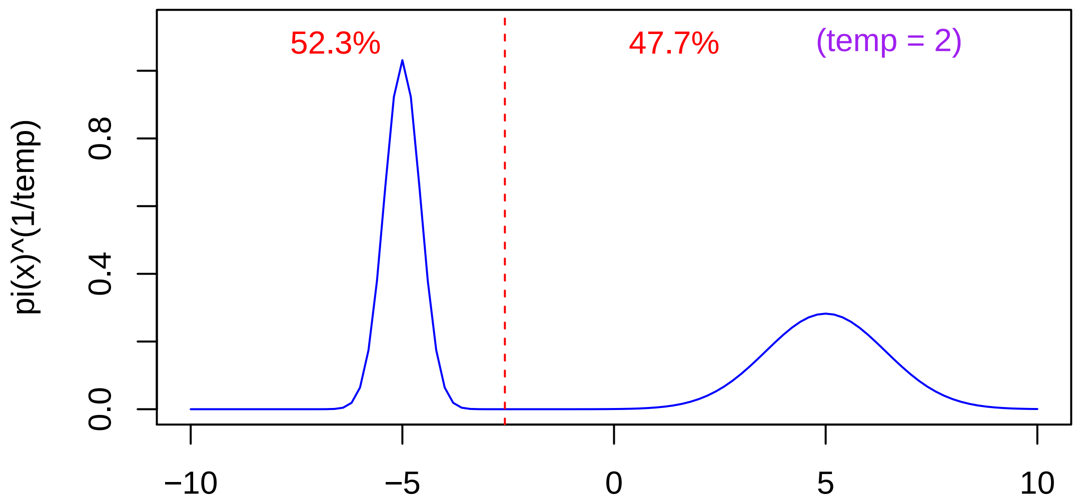


Challenge: Tempering Doesn't Preserve Mode Weights

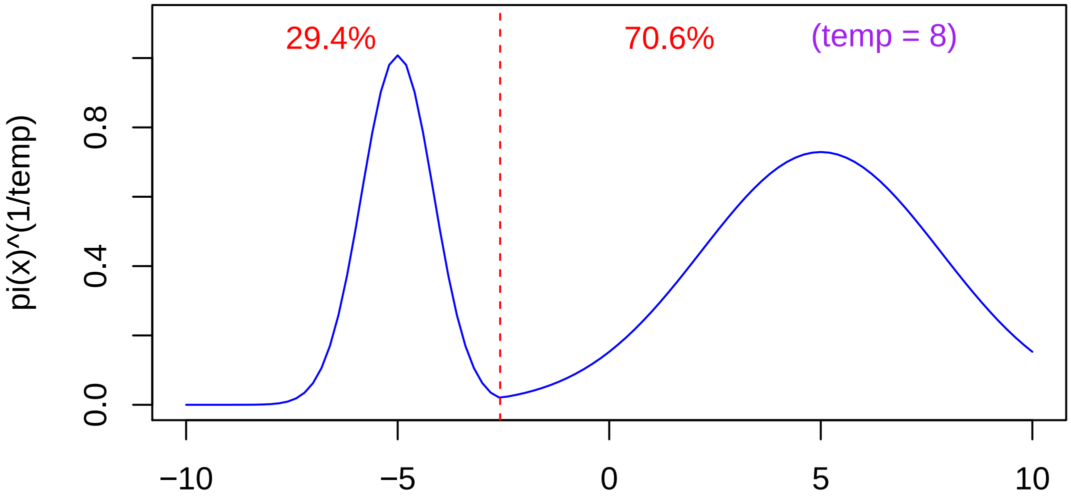
- How much "weight" (probability mass) does each mode have?
- In our example, the original ($\tau = 1$) target has a certain balance:



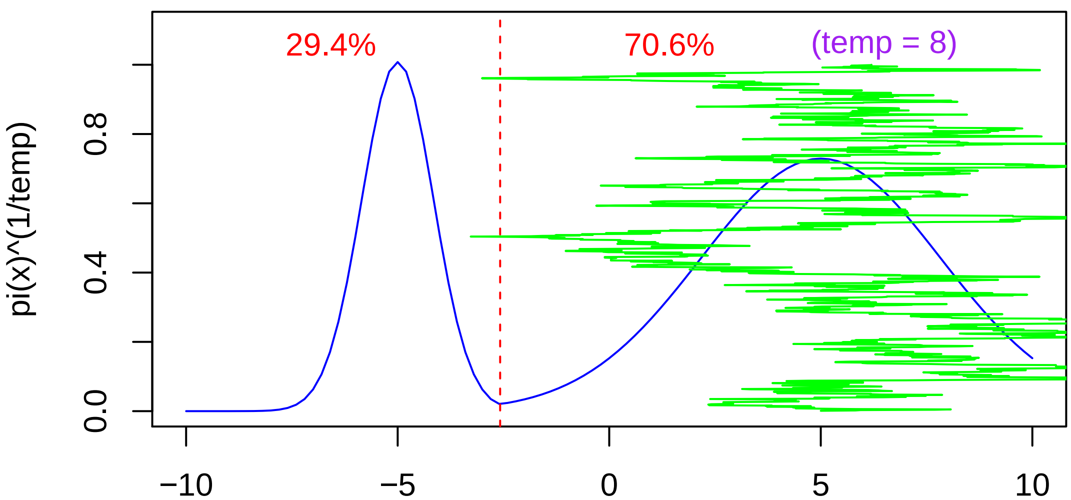
- As we do more tempering ($\tau \nearrow$), the density values get closer to 1.
- This gives more weight to "fatter" modes, even with small $\pi(x)$:



- For large enough temperatures τ, the weights become very different:



- This means that even though there are no "obstacles" to moving from 5 to -5, there is less "motivation" for the chain to do so.
- So, the chain will not move to near -5 very often.
- But, at $\tau = 1$, the mode around -5 has most of the mass of $\pi(x)$.
- In higher dimension, the weight changes become exponentially worse.
- This can lead to poor mixing (cf. Woodard et al., 2009):



- So, we have exchanged one convergence problem for another. Bad!
- (Note: I focus here on Simulated Tempering, with a <u>single</u> chain. But the same mixing problems arise for Parallel Tempering, i.e. Replica Exchange, with one chain for <u>each</u> possible temperature.)

Some Theory on Why the Weights are not Preserved

- Can we get the benefits of tempering, while avoiding weight changes?
- Suppose π is a mixture of probability distributions: $\pi(x) = \sum_j w_j \, g_j(x)$.
- Usual tempering: $\pi_\tau(x) = [\pi(x)]^{1/\tau} = \left[\sum_j w_j \, g_j(x)\right]^{1/\tau}$.
- If the components are well separated, $\pi_\tau(x) \approx \sum_j w_j^{1/\tau} \, g_j(x)^{1/\tau}$.

- Let $m_{j,\tau} = \int g_j(z)^{1/\tau}\, dz$ be the mass of $g_j(x)^{1/\tau}$. So $m_{j,1} = 1$.
- Let $f_j(x, \tau) = g_j(x)^{1/\tau} / m_{j,\tau}$ be the normalised version of $g_j^{1/\tau}$.
- Then $\pi_\tau(x) \approx \sum_j (w_j^{1/\tau} m_{j,\tau})\, f_j(x, \tau)$.
- Since $w_j^{1/\tau} m_{j,\tau} \neq w_j$ for $j \neq 1$, the weights are not preserved.
- Can we get the benefits of tempering, while avoiding weight changes?

Solution – Weight-Preserving Tempering

- Idea: Replace $\pi_\tau(x) = [\pi(x)]^{1/\tau}$ by $\pi^*_\tau(x) = [\pi(x)]^{1/\tau}\, [\pi(\mu_{x,\tau})]^{1-(1/\tau)}$.
- Here $\mu_{x,\tau}$ is the closest mode to x, at a given temperature τ.
- Then if $\pi(x) = \sum_j w_j\, g_j(x)$ are well separated, then

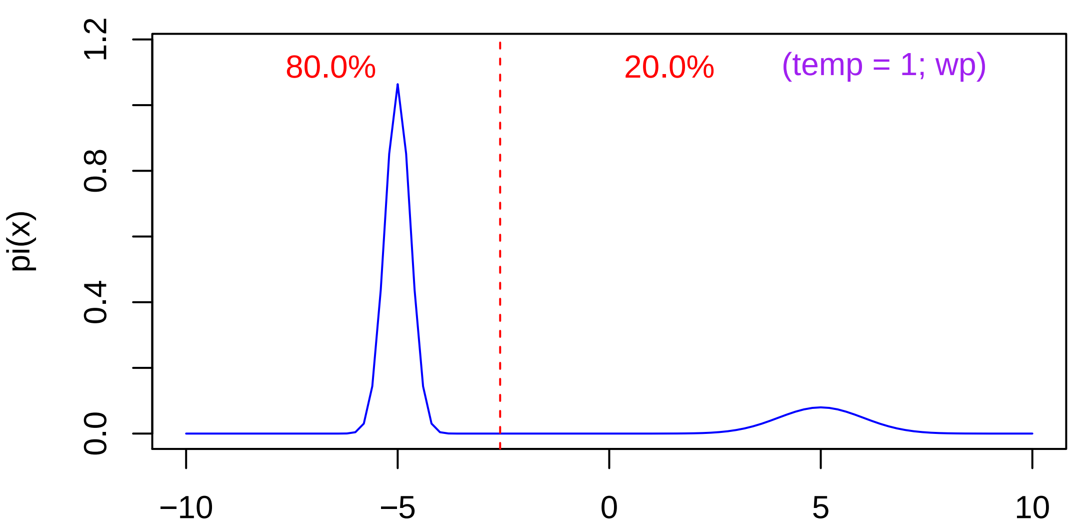
$$
\begin{aligned}
\pi^*_\tau(x) = [\pi(x)]^{1/\tau}\, [\pi(\mu_{x,\tau})]^{1-(1/\tau)} &= \Big[\sum_j w_j\, g_j(x)\Big]^{1/\tau} \Big[\sum_j w_j\, g_j(\mu_{x,\tau})\Big]^{1-(1/\tau)} \\
&\approx \Big[\sum_j w_j^{1/\tau}\, g_j(x)^{1/\tau}\Big] \Big[\sum_j w_j^{1-(1/\tau)}\, g_j(\mu_{x,\tau})^{1-(1/\tau)}\Big] \\
&\approx \sum_j \Big[w_j^{1/\tau}\, g_j(x)^{1/\tau}\Big] \Big[w_j^{1-(1/\tau)}\, g_j(\mu_{x,\tau})^{1-(1/\tau)}\Big] \\
&= \sum_j w_j\, g_j(x)^{1/\tau} g_j(\mu_{x,\tau})^{1-(1/\tau)}.
\end{aligned}
$$

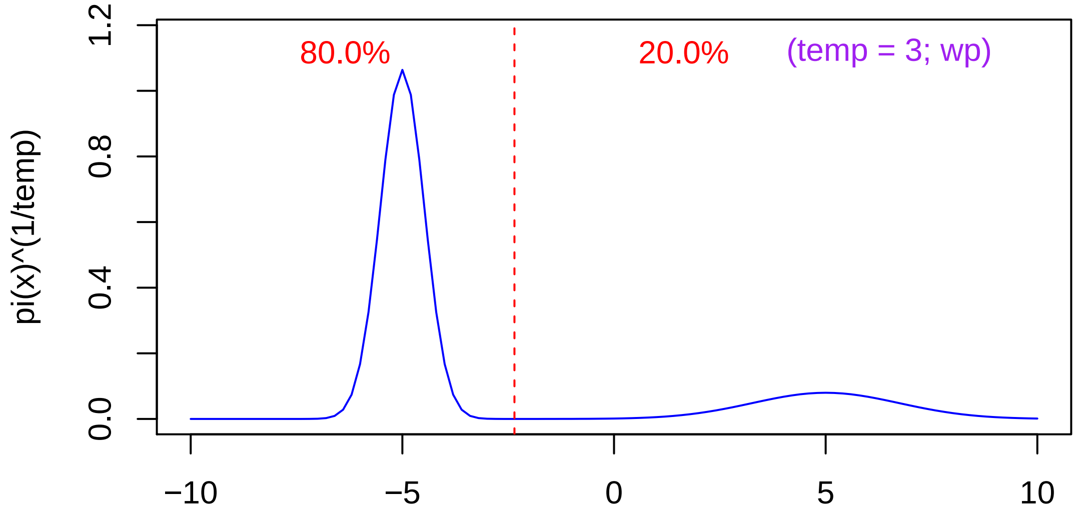
- Near the mode, $g_j(x)^{1/\tau} g_j(\mu_{x,\tau})^{1-(1/\tau)} \approx g_j(x)^{1/\tau} g_j(x)^{1-(1/\tau)} = g_j(x)$,

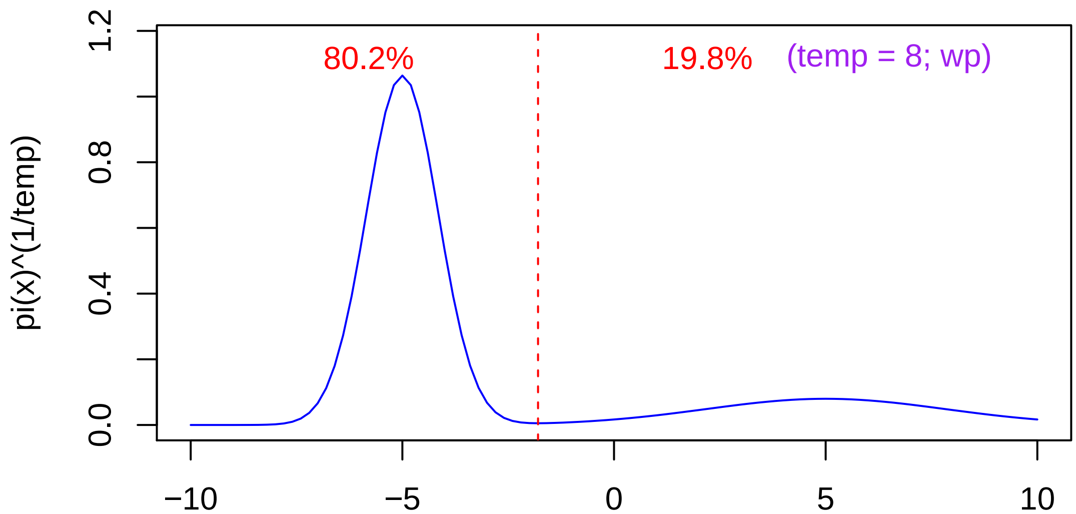
so $\int g_j(x)^{1/\tau} g_j(\mu_{x,\tau})^{1-(1/\tau)} dx \approx 1$, so mode j has weight $\approx w_j$. Phew!
- For example, in the Gaussian case where $g_j(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/2\sigma^2}$,

$$
\int g_j(x)^{1/\tau} g_j(\mu)^{1-(1/\tau)} dx = \int \Big(\frac{1}{\sqrt{2\pi}\sigma}\Big)^{1/\tau} e^{-(x-\mu)^2/2\sigma^2\tau} \Big(\frac{1}{\sqrt{2\pi}\sigma}\Big)^{1-(1/\tau)} dx = \sqrt{\tau}
$$

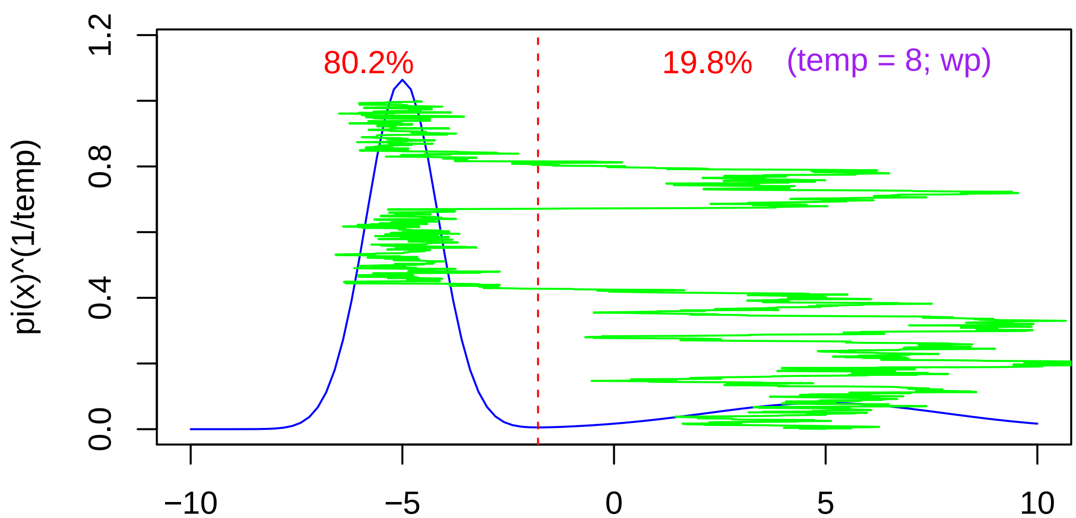
which depends only on τ (not σ), so weight ratios are preserved. Good!
- Let's try this π^* on our example, for different temperatures:







- Weights are approximately preserved. But still mixes pretty well:



- THEOREM: Under certain (strong) assumptions, mixing time is $O[d\,(\log d)^2]$ in dimension d. Works well in simulations, too. Good!
- Apply to discrete distributions, like DA? Maybe – let's discuss it!