CUED - Engineering Tripos Part IIB 2019-2020 Module Coursework

Module	**4F13**	Title of report	**Probabilistic Ranking**

Date submitted: **19/11/2020**	Assessment for this module is ☑ **100%** / ☐ 25% coursework of which this assignment forms **33** %

UNDERGRADUATE STUDENTS O NLY			**POST GRADUATE STUDENTS ONLY**			
Candidate number:	**5562E**		Name:		College:	

Feedback to the student

☐ **See also comments in the text**

		Very good	**Good**	Needs improvmt
C O N T E N T	**Completeness, quantity of content:** Has the report covered all aspects of the lab? Has the analysis been carried out thoroughly?			
	Correctness, quality of content Is the data correct? Is the analysis of the data correct? Are the conclusions correct?			
	Depth of understanding, quality of discussion Does the report show a good technical understanding? Have all the relevant conclusions been drawn?			
	Comments:			
P R E S E N T A T I O N	**Attention to detail, typesetting and typographical errors** Is the report free of typographical errors? Are the figures/tables/references presented professionally?			
	Comments:			

Overall assessment (circle grade)	A*	**A**	B	C	D
Guideline standard	>75%	**65-75%**	55-65%	40-55%	<40%
Penalty for lateness:		*20% of marks per week or part week that the work is late.*			

Marker: Date:

4F13 Probabilistic Machine Learning - Probabilistic Ranking

Candidate: 5562E

November 19, 2020

Contents

1 Questions

a Gibbs Sampling

Gibbs sampling is used to sample from a multi-variate distribution by sequentially sampling from univariate conditionals. In the True-Skill model, we can approximate the conditional distribution with a 1-D Gaussian of a certain mean and variance - which are functions of the other variables and game outcomes. A portion of the modified code to compute these samples is given in listing 1.

Listing 1: Gibbs sampling additions

```
m = np.zeros((M, 1))
for p in range(M):
        # fill in m[p] prediction (natural param conditional)
        wins_array = np.array(G[:, 0] == p).astype(int)
        loss_array = np.array(G[:, 1] == p).astype(int)
        m[p] = np.dot(t[:,0], (wins_array - loss_array))
```

We plot the sampled player skills for a few players (figure 1). These data do appear noisy (in some sense random) but it is hard to tell how soon the Gibbs sampler transitions into a stable probability density region.

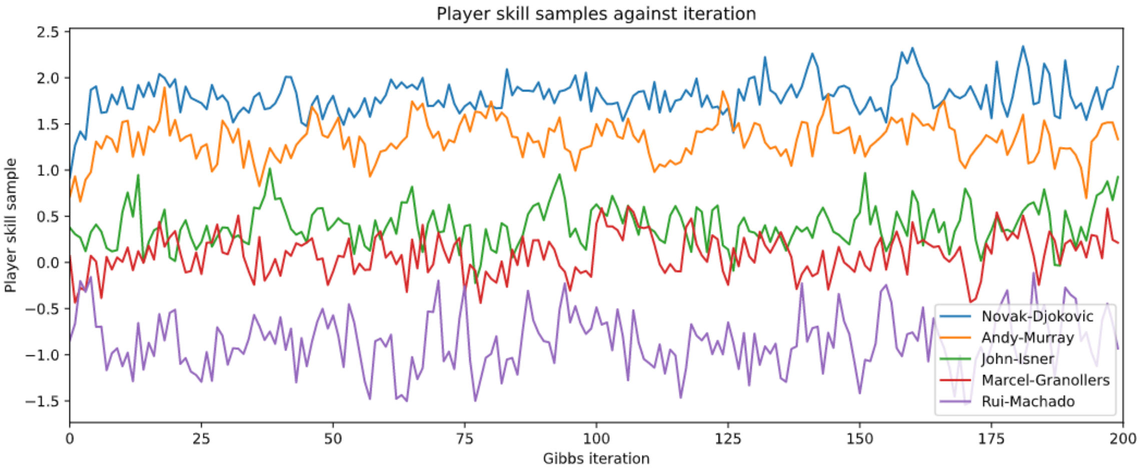


Figure 1: Gibbs skill samples for various players

To investigate the burn-in time, we plot the population mean and standard deviation at each Gibbs iteration - figure 2. The population mean gives us little information but we see that the standard deviation converges to a steady value fairly quickly (only after 10 iterations). We set the burn-in time to $b = 10$ and discard all preceding samples.

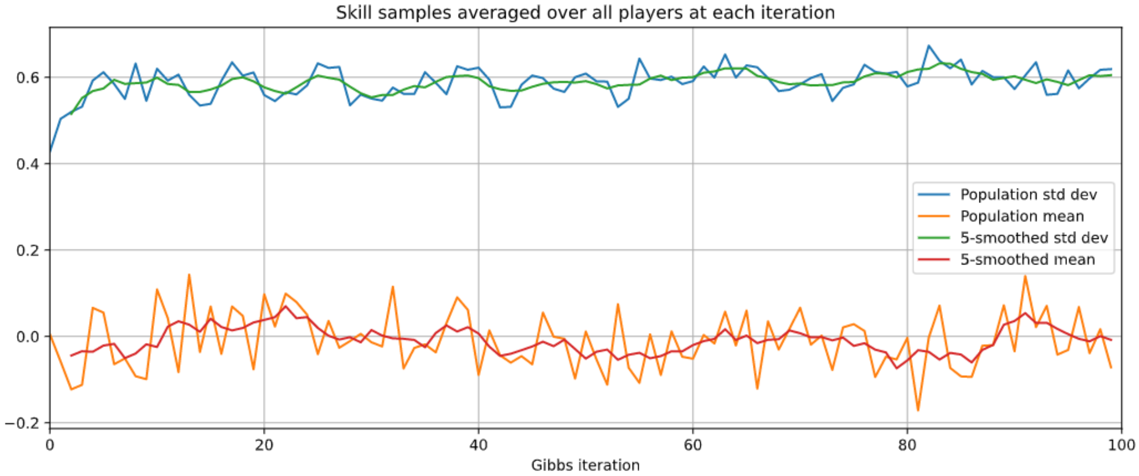


Figure 2: Gibbs skill samples averaged over all players at each iteration

However, there is an additional wrinkle, the plot in figure 1 is smoother than it should be; neighbouring samples are not independent. To test this hypothesis, we plot the auto-correlation of each player's skill samples (figure 3). For samples to be approximately independent, their correlation should be close to 0. We get good enough for an offset of $t = 5$ - excepting one player: Djokovic.

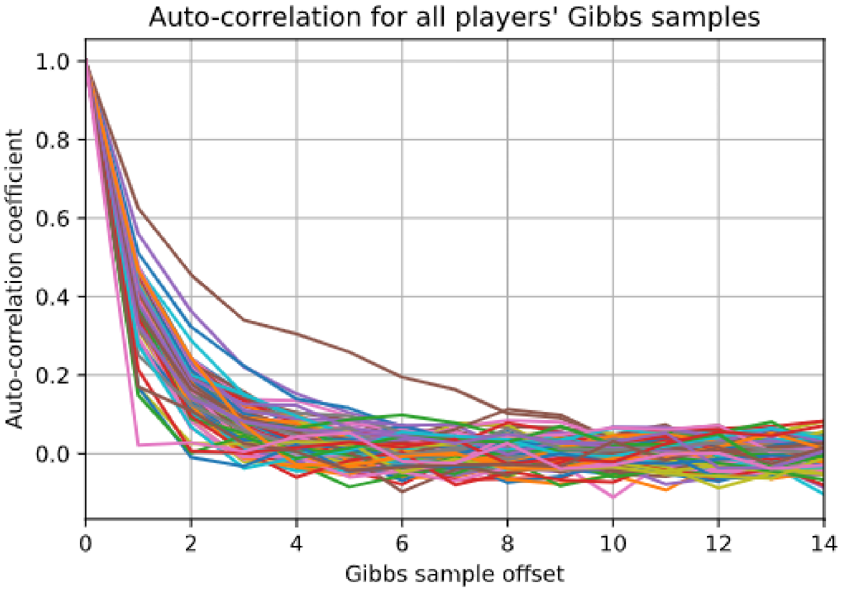


Figure 3: Auto-correlation for Gibbs sampler for all players

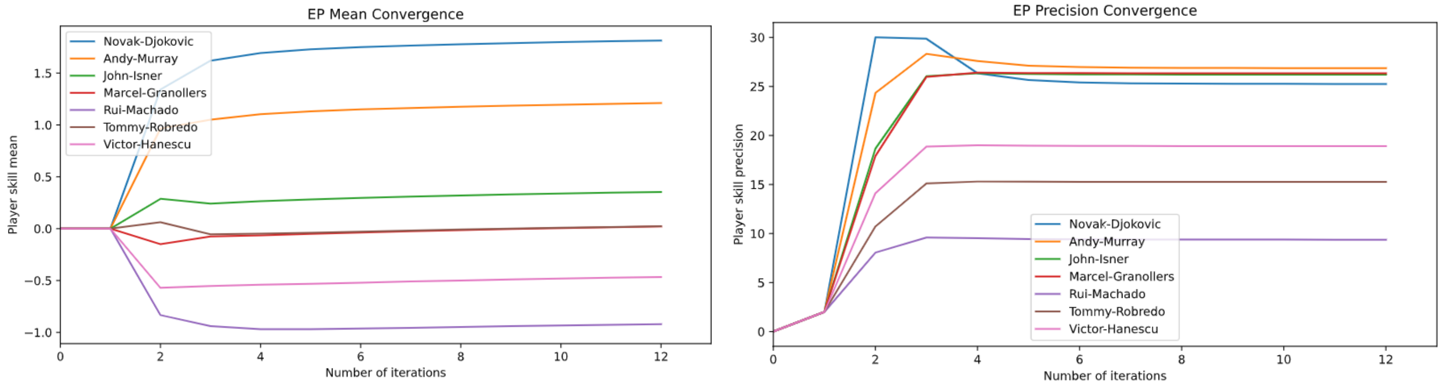
In summary, to run our Gibbs sampler reliably, we only use samples after a burn-in period of $b = 10$ iterations. Even then, we thin the samples such that we only keep every t'th ($t = 5$) sample to ensure that they are approximately independent. We use these values $b = 10$ and $t = 5$ for the rest of the report.

b EP - Message Passing

Nevertheless, Gibbs sampling is rather computationally intensive. An alternative is message-passing through the Expectation Propagation (EP algorithm). This treats each player as a vertex on a graph, connected by directed edges representing match outcomes. We can use message passing (a form of Belief Propagation) to iteratively improve on our guess for each player's marginal skill. We approximate the marginal skill by a Gaussian with two parameters: mean and precision (inverse variance). These parameters have the advantage of combining simply to form the messages.

In Gibbs sampling, we run a Markov chain and aim to converge to a stationary probability distribution. This target distribution is the joint player skill distribution. However, in EP we do not converge to a distribution but rather a stable graph object; further iterations do not alter the parameters of each vertex: mean and precision.

In the previous section, we saw the Gibbs sampler only takes $b = 10$ samples to converge to a stationary distribution. For the EP algorithm, we can plot the player parameters against iteration number for a few players to gauge convergence (figure 4).



(a) EP player skill mean evolution

(b) EP player skill precision evolution

Figure 4: EP player skill parameter evolution against iteration

We converge quickly for this set of players. Setting the number of iterations to $T = 5$ seems reasonable to get a good estimate of the rankings as the mean and precision do not change much after this point. This value of $T = 5$ will be used throughout the rest of the report.

c EP - Player Comparison

Suppose that a player i goes up against j. We denote their skills by $w_i \sim \mathcal{N}(m_i, \sigma_i^2)$ and $w_j \sim \mathcal{N}(m_j, \sigma_j^2)$ respectively. The mean and variance parameters (inverse precision) are estimated through EP.

The performance difference $s_{ij} = w_i - w_j$ is corrupted by Gaussian noise of unit variance $t_{ij} = s_{ij} + n$ to account for performance inconsistency ($n \sim \mathcal{N}(0, 1)$). The match result is then given by the sign of t_{ij} such that $t_{ij} > 0 \Rightarrow i$ wins and j wins otherwise.

Having run EP, we wish to compute the probability that i is more skilful than j: $P(s_{ij} > 0)$. This is different from the probability that i beats j in a head-to-head: $P(t_{ij} > 0)$. These probabilities are easy to compute by assuming that $w_i \perp\!\!\!\perp w_j \perp\!\!\!\perp n \perp\!\!\!\perp w_i$, so means and variances combine additively:

$$s_{ij} = w_i - w_j \sim \mathcal{N}(m_i - m_j, \sigma_i^2 + \sigma_j^2) \tag{1}$$

$$t_{ij} = s_{ij} + n \sim \mathcal{N}(m_i - m_j, \sigma_i^2 + \sigma_j^2 + 1) \tag{2}$$

For any Gaussian r.v. $X \sim \mathcal{N}(\mu, \sigma^2)$, $P(X > 0) = \Phi(\mu/\sigma)$ - where $\Phi(\cdot)$ is the standard Gaussian c.d.f. Applying this result to equations 1 and 2 for the top four ranked players, we obtain table 1.

$P(s_{ij} > 0)$	**Djokovic**	**Federer**	**Nadal**	**Murray**
Djokovic	-	0.91	0.94	0.99
Federer	0.09	-	0.58	0.81
Nadal	0.06	0.42	-	0.76
Murray	0.01	0.19	0.24	-

(a) Prob. row player is more skilful

$P(t_{ij} > 0)$	**Djokovic**	**Federer**	**Nadal**	**Murray**
Djokovic	-	0.64	0.66	0.72
Federer	0.36	-	0.52	0.59
Nadal	0.34	0.48	-	0.57
Murray	0.28	0.41	0.43	-

(b) Prob. row player wins a head-to-head

Table 1: Top four players comparison based on EP (5 iterations)

We are more confident in who is better than in who will win: $|P(s_{ij} > 0) - 0.5| > |P(t_{ij}) - 0.5| \quad \forall i, j$. This makes sense mathematically as the variance $\sigma_t^2 > \sigma_s^2$. It also makes sense intuitively because an individual match is subject to performance variation. I would bet my house that Djokovic is a better tennis player than Andy Murray, but I'm not betting that much on Djokovic winning his match against him tomorrow.

d Gibbs - Player Comparison

However, we can also compare players through Gibbs sampling. Restricting ourselves to just Djokovic and Nadal, we compare them in three ways:

1. Approximate marginals by Gaussian
2. Approximate joint by bivariate Gaussian
3. Empirical comparison of joint samples

The first method is shown in figure 5. We estimate the parameters of each Gaussian by taking the mean and variance of each player's Gibbs samples (after burn-in and thinning). Indeed a Gaussian does seem to be a good fit for the data. We see that Djokovic has higher mean and lower variance.

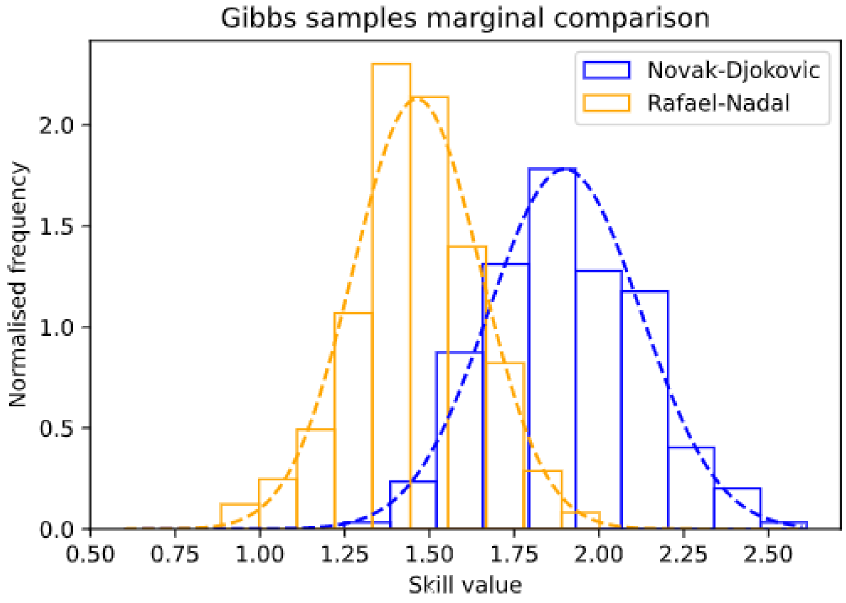


Figure 5: Gibbs samples marginal comparison - Djokovic v Nadal

We can also fit a bivariate Gaussian to the pair-wise samples $\{(w_i^{(n)}, w_j^{(n)})\}_{n=1}^N$ (figure 6). We approximate the parameters as before but now noting that we have an additional covariance term σ_{ij}^2 off the diagonal of the 2×2 variance matrix. This covariance term is non-zero (in fact positive) meaning that the principal axes of the Gaussian are slightly skewed.

The last method involves simply plotting the difference of the two skill samples (figure 7). Any bars coloured red have Nadal higher skill than Djokovic.

Each of these methods give a different way of estimating the probability player i is more skilful than j: $P(s_{ij} > 0)$. They are listed below:

1. Independent Gaussians - $\Phi((m_i - m_j)/(\sigma_i^2 + \sigma_j^2))$
2. Correlated Gaussians - $\Phi((m_i - m_j)/(\sigma_i^2 - 2\sigma_{ij}^2 + \sigma_j^2))$
3. Empirical comparison - $N^{-1} \sum_{n=1}^N \mathbb{1}(w_i^{(n)} > w_j^{(n)})$

Figure 6 shows that the samples are not independent so method 2 is preferred over 1. In addition, as the probabilities can be very small, the empirical estimate (though unbiased) will need a very high number of samples N to be accurate (the red area in figure 7 is very small). Therefore we prefer method 2 for comparing player skills. Doing this for the top four players yields table 2.

Table 2 agrees with 1a very well. If anything the predictions tend to be more confident for the joint Gaussian model as the covariance between each of the top four players' skills is positive. Therefore, this decreases the variance of the difference, leading to more confident predictions.

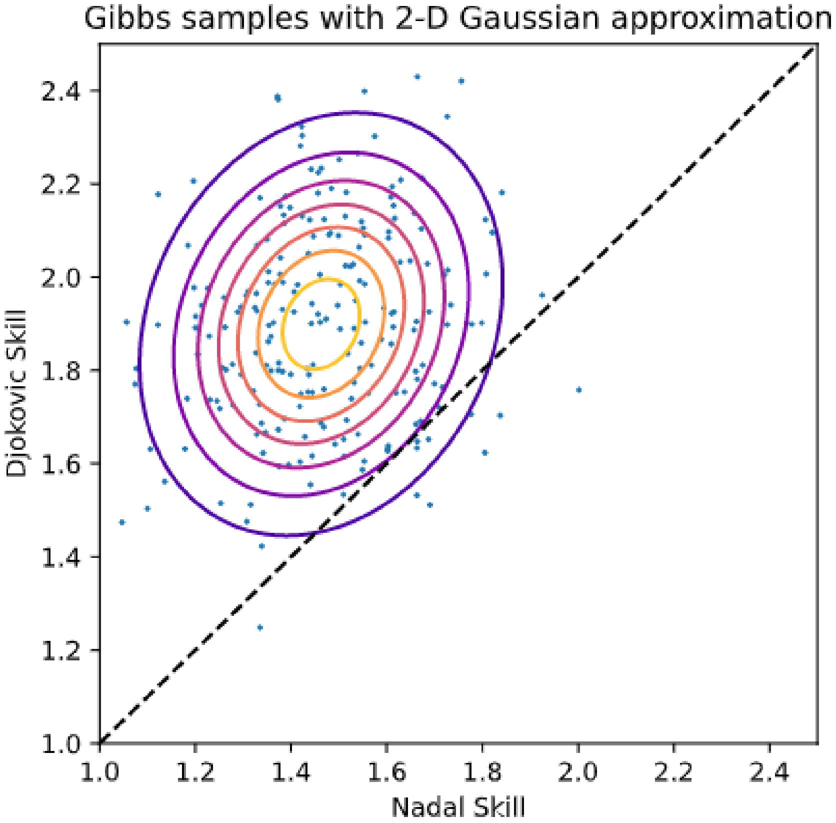


Figure 6: Gibbs samples joint comparison - Djokovic v Nadal

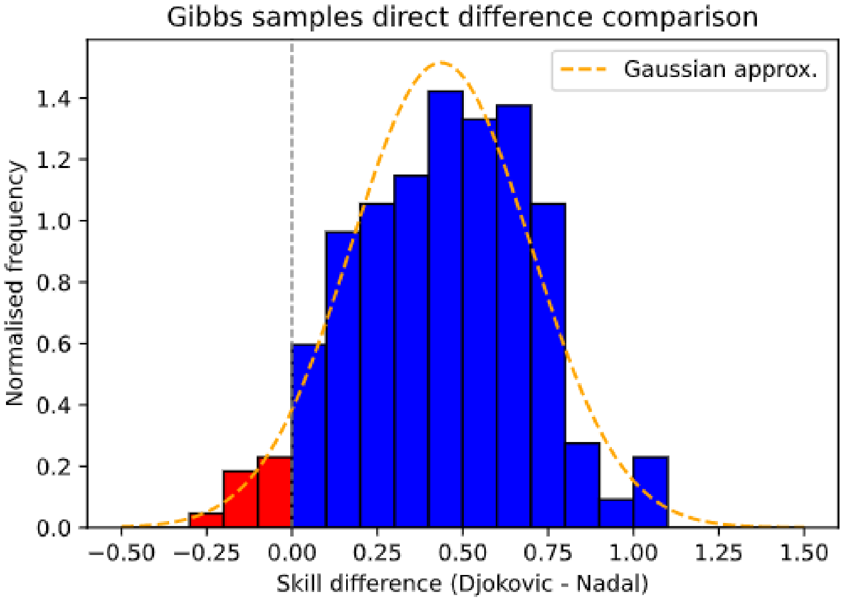


Figure 7: Gibbs samples direct difference comparison - Djokovic v Nadal

$P(s_{ij} > 0)$	**Djokovic**	**Federer**	**Nadal**	**Murray**
Djokovic	-	0.92	0.95	0.98
Federer	0.08	-	0.63	0.81
Nadal	0.05	0.37	-	0.75
Murray	0.02	0.19	0.25	-

Table 2: Player skills head-to-head - $P(s_{ij} > 0) = \Phi((m_i - m_j)/(\sigma_i^2 - 2\sigma_{ij}^2 + \sigma_j^2))$

e Method Comparison: Win ratio, Gibbs and EP

We go about ranking players using different methods: win ratio; Gibbs sampling for True-Skill; and EP estimates for True-Skill.

The win ratio model assumes that each player has a fixed probability of winning their match irrespective of their opponent. If a player has played k out of n matches, their win ratio is distributed as $r \sim \mathcal{N}(k/n, k(n-k)/n^2)$; this applies standard results for combining n independent Bernoulli r.v.'s.

With Gibbs sampling, we sample the player skill from conditionals such that we mimic sampling from the joint. After burn-in and thinning, we have a set of independent samples for the skill of each player and we can compute the mean and variance of this set empirically.

For EP, we run the message-passing for 5 iterations to achieve convergence. This returns the expected means and precisions for the skill of each player.

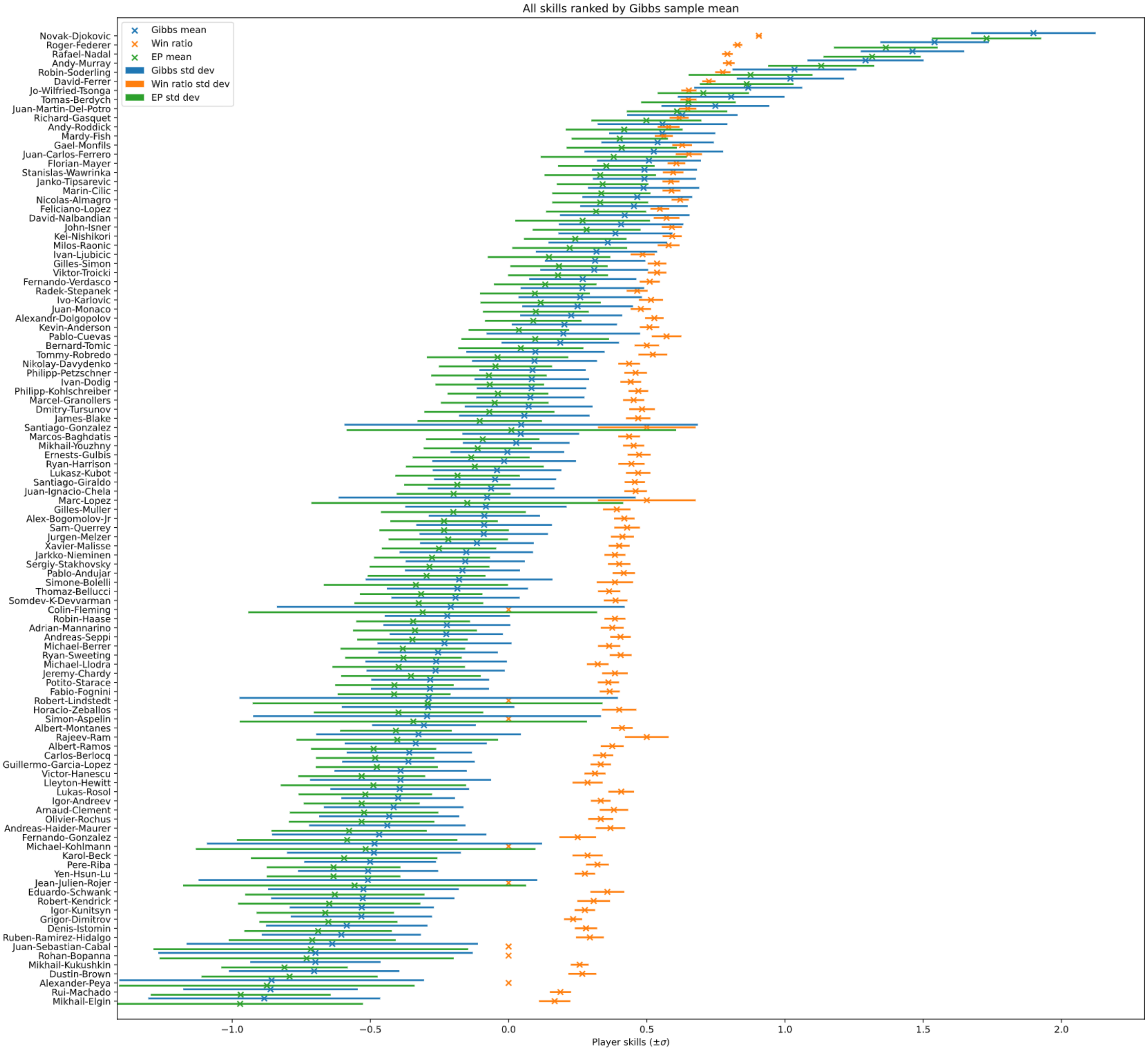


Figure 8: Player skills ranked by Gibbs sample mean

We plot all three methods on figure 8 - rank ordering by Gibbs sample mean. Gibbs and EP match closely. The means seem to differ only by a constant and a small amount of noise. The constant offset does not matter when comparing players but it is determined by the prior $p(w) \sim \mathcal{N}(0,1)$. It is possible that the EP algorithm drifts away from this prior after initialisation. The noise we see (slight discrepancies in rank order) may be due to approximation errors from each approach. Nevertheless, the variances of both Gibbs and EP match closely.

As for the win ratio, its rank ordering does not agree especially well with the other two. This is expected as it ignores the opposition. Nonetheless, in general higher win ratio means higher skill. There are several players with a win ratio of 0 who are moderately high up on the Gibbs rankings (e.g. Colin Fleming). These players must have played very few games (losing all of them) and so their mean stays close to the prior mean of 0 albeit with high variance. If we increased the prior variance, we would see fewer players with no wins to their name appearing so high up in the rankings.

Words: 997