⚠️ NOTA ⚠️

⚠️ Riguardo alle varie aggiunte manderò alla fine un pdf con sottolineate/evidenziate bene le parti nuove.
⚠️ perché magari quello che riporto qui (ciò che è nuovo è **in grassetto**) in questo file lo aggiornerò riscrivendo la tesi, per esempio specialmente per sistemare l'inglese
⚠️ quindi questo è solo un file veloce per riassumere le mie risposte alle sue annotazioni, non una cosa definitiva

Nota 1

- [x] In Figura 1.6 c'è un errore nella legenda, il termine "cluster 4" è ripetuto due volte.

Sistemato, era un errore dai copia incolla in julia.

Nota 2

- [x] Nella discussione alla fine del capitolo 1, aggiungerei come il fatto di standardizzare le covariate da utilizzare per il clustering può aiutare a scegliere parametri più "standard" per g4.

Ho fatto questa breve aggiunta per ora:

> On the other hand, similarity g_4 appeared to be quite sensitive to the parameters regulating the $\invgamma$ distribution for σ^2. **This suggests that covariates should be standardized prior to their usage, in order to simplify the research and uniform the choice of the most appropriate set of parameters.** In any case, the JDRPM implementation can provide some flexibility in this regard by possibly assigning separately the pair of a_0 and b_0 for each covariate that we wish to include in the clustering process.

Nota 3

- [x] Listing 2: perché non salvarti alcuni valori fissi, come lgamma(a_c), a_c*log(b_c) e così via? Credo possano risultare in migliori performance in cambio di poca memoria salvata.

Giusto. Il suo impatto è magari minimo ma contando il numero di chiamate alle similarità/coesioni sul lungo termine farà risparmiare. È una modifica forse solo secondaria per ora, nel senso utile in futuro a chi userà il codice ma meno ora che ho già svolto tutti i test, quindi penso la implementerò dopo la fine delle scadenze. Perché chiamando le coesioni e similarity in modo indiretto, tramite funzioni di questo tipo

```
function covariate_similarity!(idx::Real, X_jt::AbstractVector{<:Real}, cv_params::Vector, R::Real, lg::Bool,
case::Int=1, add::Bool=false, lS=@MVector(zeros(2)), cv_weight::Real=1)
    if idx==1 similarity1!(X_jt,cv_params[1],lg,case,add,lS,cv_weight); return; end
    if idx==2 similarity2!(X_jt,cv_params[1],cv_params[2],lg,case,add,lS,cv_weight); return; end
    if idx==3 similarity3!(X_jt,cv_params[1],cv_params[2],lg,case,add,lS,cv_weight); return; end
    if idx==4 similarity4!(X_jt,cv_params[1],cv_params[2],cv_params[3],cv_params[4],lg,case,add,lS,cv_weight);
return; end
end
```

diventa poi un po' meno immediato fare modifiche, quindi ora forse devo concentrarmi solo sullo scrivere bene la tesi.

Nota 4

- [x] In Sezione 3.1.1, non riesci ad ottenere esattamente lo stesso clustering aumentando il numero di iterazioni e di burnin? Inoltre, toglierei le informazioni riguardanti MSE, LPML e WAIC, lasciando solo l'execution time: il modello utilizzato, da un punto di vista teorico, è quasi uguale a quello originale di Page et al., cambia solo la scelta delle prior (InvGamma al posto di Uniform). Quindi, il miglioramento ottenuto nelle performance sembra essere più un risultato del caso piuttosto che il risultato di una nostra scelta modellistica. Per lo stesso motivo, le toglierei anche dalla sezione 3.1.2.

Per le metriche giusto, tolgo. Per i fit ho leggermente sistemato, aggiungendo iterazioni e testando con vari parametri. Ora sono più simili ma non sembrano mai volersi accordare perfettamente. Comunque avevo messo (e ho ora agggionato) dei commenti riguardo a perché entrambe le soluzioni fossero ragionevoli, e non essendo un cluster “esatto” forse ci sta che escano leggermente diverse.

Se vuoi vedere le modifiche ai fit e cluster ho già aggiornato il file su overleaf, con per esempio i nuovi plot. Ma comunque li riporto anche qui.

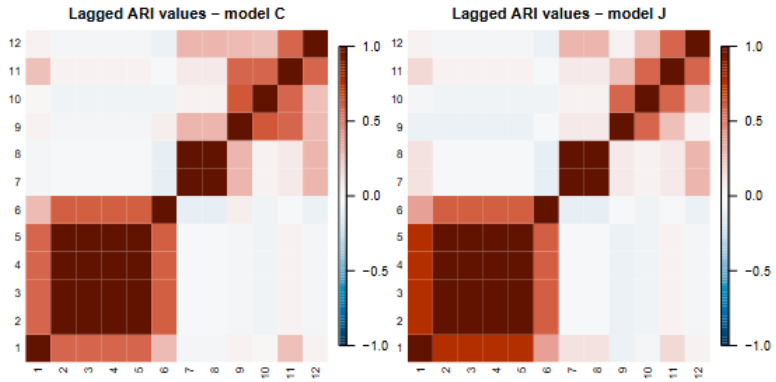


Figure 3.1: Lagged ARI values for CDRPM and JDRPM in the first simulated scenario.

Figure 3.2: Clusters estimates produced by CDRPM and JDRPM in the first simulated scenario. Time instants are on the x axis, units are indicated vertically by their number, and colors represent the cluster label.

Ho fatto anche questa aggiunta per esempio (da sistemare come le altre aggiunte riportate qui in questo file):

> At this testing stage there were just the target values from Y_{it} to dictate the clusters definition, and both models managed to provide good fits, as we can read from Table \ref{tab: fits metrics no space}. We can see how JDRPM obtained a faster execution time, which is however not really relevant in this small-sized fit. Regarding the fitted values, displayed in Figure \ref{fig: fitted and target values no space} together with the original data, they turned out to be also more precise than the one generated by CDRPM, having lower mean squared errors. **We do not report the fitting metrics of WAIC and LPML since the models were conceptually equivalent and tested under the same hyperparameters and MCMC iteration setup. Therefore, any observed difference in these metrics could be likely attributed to chance.**

Nota 5

- [ ] In vari punti del testo ho visto che hai scritto “trough”, invece dovrebbe essere “through”. Infatti, con “trough” si intende una mangiatoia per animali (https://www.dictionary.com/browse/trough (https://www.dictionary.com/browse/trough)), che non credo rientri nei nostri discorsi.
- [ ] Alla fine di pagina 33 c’è un “helps to emphasizes”, va corretto in “helps to emphasize”.

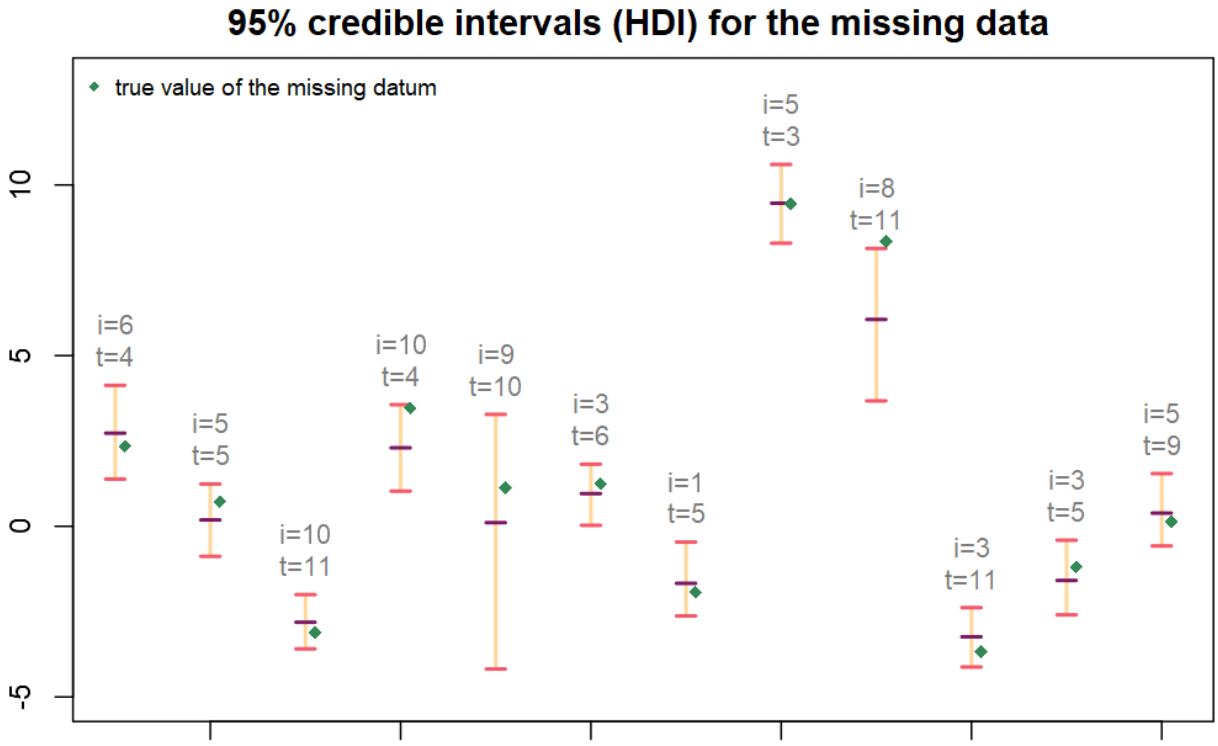
Si qui mi spiace se ti/vi ho fatto spulciare ogni parola, pensavo che la vostra correzione si riferisse solo ai concetti, mentre la sintassi e gli errori sintattici li avrei naturalmente ricontrollati io alla fine. In effetti però la *trough* mi sarebbe quasi certamente sfuggita :)

Nota 6

- [ ] Nelle sezioni 3.2.1 e 3.2.2 sarebbe carino mostrare i credible intervals nei punti in cui mancano i dati, in questo modo possiamo verificare se la vera osservazione cade nel credible interval. Non sono sicuro che mostrarlo per tutti i punti mancanti sia fattibile, magari nel caso possiamo mostrarlo per pochi esempi (magari alcuni che funzionano e altri che non funzionano).

Per i fit NA senza spazio li ho aggiunti ora. Ho dovuto cambiare gli NA, e quindi rifare i fit, perché quando li avevo fatti la prima volta non mi ero salvato i seed, quindi mi ero perso quali indici fossero stati assegnati a NA. Invece ora è tutto più preciso e coerente.

Il grafico dei CI per il test NA sarebbe questo:



Nota 6 - Kriging

Un'altra cosa che potresti fare è mettere a NA un'intera time series e plottare media e credible intervals ottenuti per quella time series, così da vedere quanto il modello è bravo a predirre per un punto nello spazio in cui non potrebbe non esserci un sensore. Per intenderci, come in Figura 2 e Figura 3 a pagina 1013 di questo documento: https://it.pearson.com/content/dam/region-core/italy/pearson-italy/pdf/Docenti/Universit%C3%A0/bozza-book-compresso-new1.pdf (https://it.pearson.com/content/dam/region-core/italy/pearson-italy/pdf/Docenti/Universit%C3%A0/bozza-book-compresso-new1.pdf). Questa cosa potrebbe essere fatta prendendo un punto e poi facendo vedere gli intervalli nel caso del modello base, poi con il clustering spaziale, poi con le covariate in likelihood e/o clustering. Si suppone che aumentando la complessità del modello piano piano ci sia un miglioramento nella predizione.

Mi piace molto questa idea, e l'ho fatta ora, volevo solo un consiglio su alcune cose. Inoltre, per capire dove inserire questa parte, ho scritto la mia idea alla Nota 10.

1. Un titolo migliore per questo capitolo. All'inizio era Testing, ma la prof ha suggerito l'altro che vedi in cima alla testatina della pagina (Comparing the two algorithms over simulated examples and a real world application). Secondo me è un po' troppo prolisso, volevo chiederti se tu avessi qualche idea migliore.
2. Se questo formato di plot va bene, o se invece avessi qualche idea migliore su come proporlo.

Idea originale ma forse troppo distesa

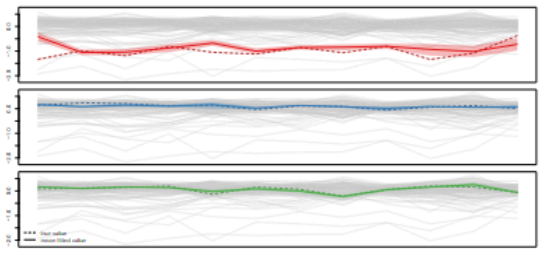


Figure 3.34: Enter Caption

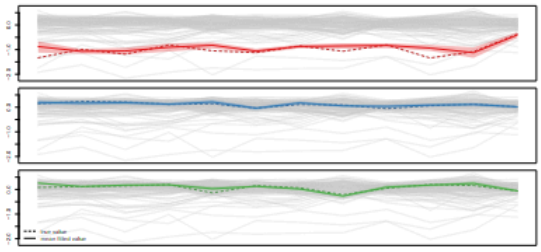


Figure 3.35: Enter Caption

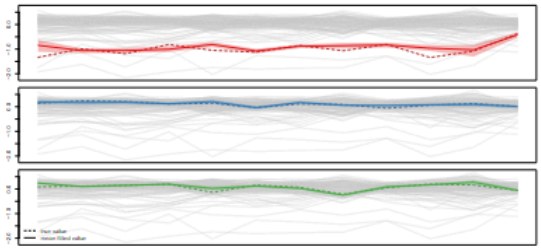


Figure 3.36: Enter Caption

Questa penso meglio (devo sistemare gli axis ticks ecc, ma non era quello l'importante ora)

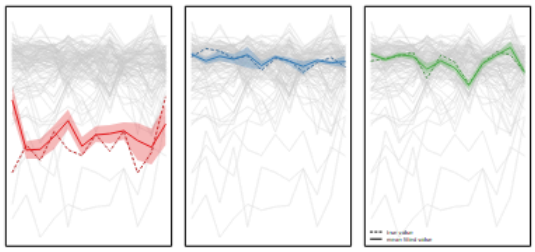


Figure 3.37: Enter Caption

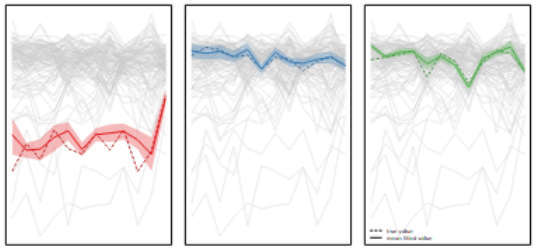


Figure 3.38: Enter Caption

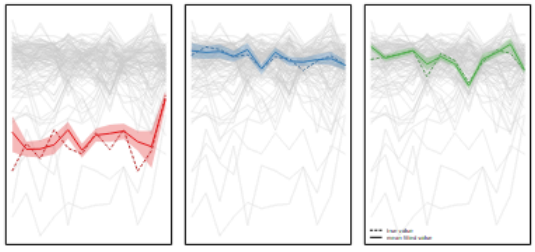
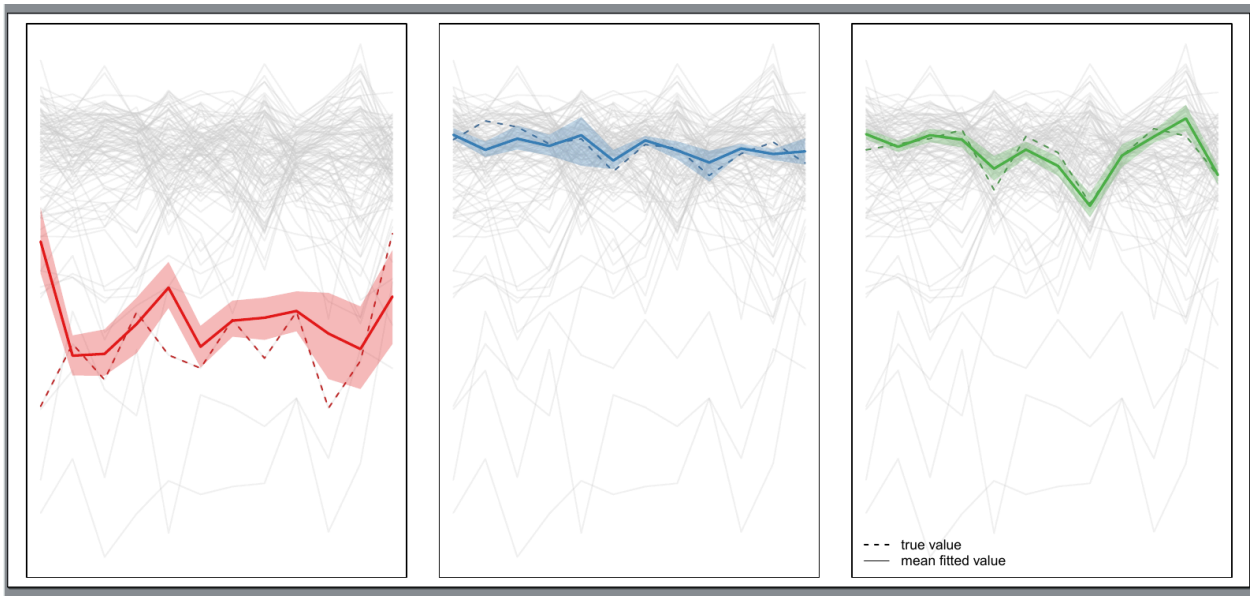


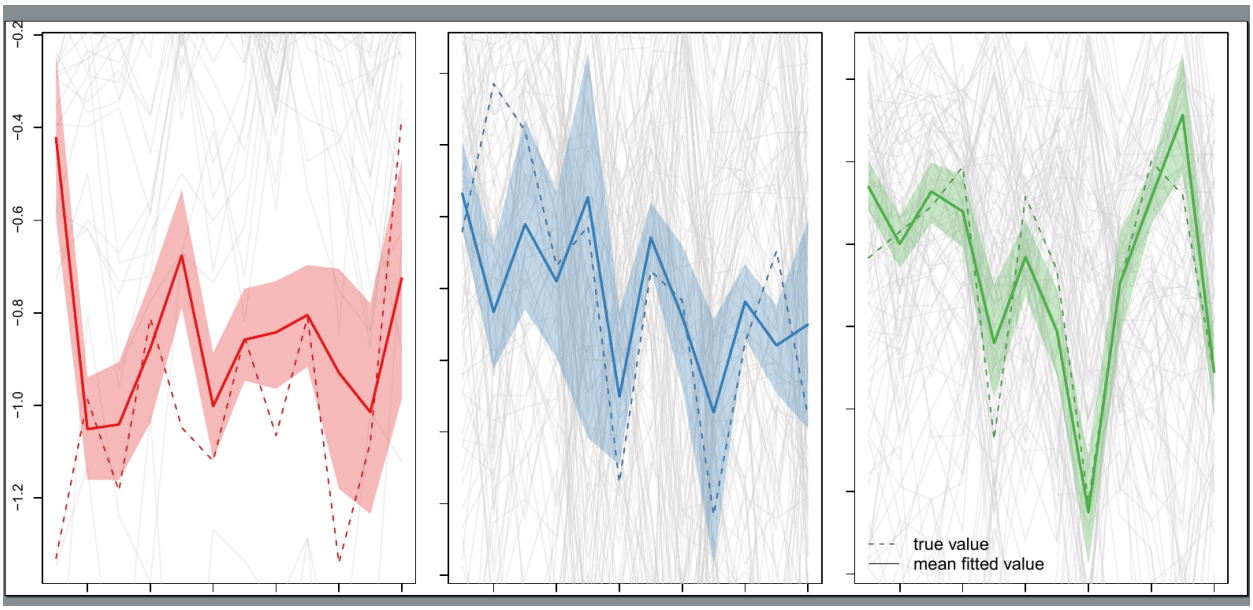
Figure 3.39: Enter Caption

Ci sono tre plot perché corrispondono a fit JDRPM space, space + Xlk, space + Xlk + Xcl, in quest'ordine dall'alto verso il basso. Non ci sono titoli perché la prof non li voleva, vuole che metta tutto nelle caption.

altrimenti rispetto a queste versioni di plot “normali”



c’è anche questa che fa uno zoom/focus sulle missing time series, tagliando come si vede il resto delle osservazioni in grigio che quindi sforano un po’ dai bordi del plot



Nota 7

- [x] Sezione 3.3: a questo punto, standardizzerei direttamente le covariate. In questo modo, hai anche una maggiore interpretabilità dei beta che ottieni.

Qui dovrei aver già sistemato. Forse aveva fatto questa nota in una vecchia versione del documento, in cui in effetti avevo riportato dei risultati senza standardizzare le covariate. Ora/dopo un qualche aggiornamento avevo invece usato ovunque covariate standardizzate. Infatti ho messo come premessa alla sezione 3.3 (che comunque diventerà 2.3 perché la prof mi ha detto di far partire la numerazione 1 dal capitolo description of the model):

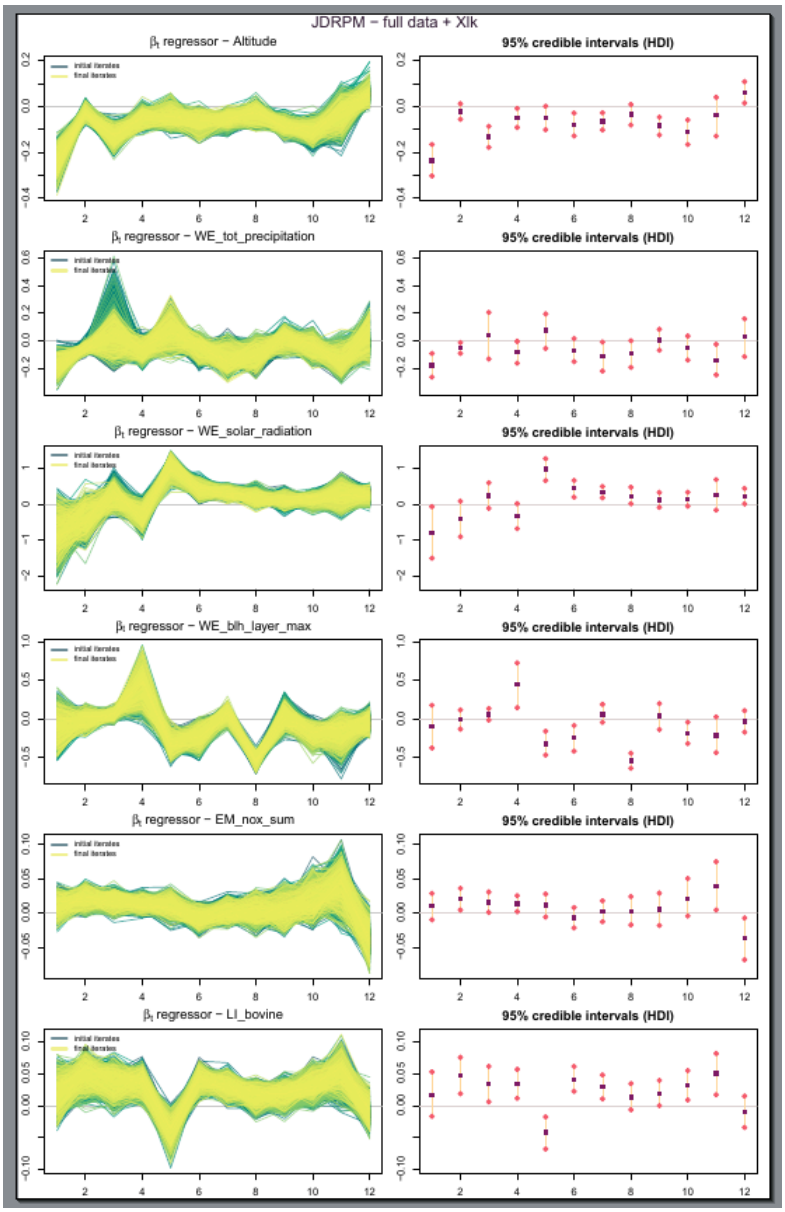
> **To perform the fits that will now follow, all the included covariates were treated in the same way as the target value, with the time-wise centering procedure and reasoning described in Section 3.1.2.**
> Figure 3.15 provides an exemplification of the effect of this transformation on one of the covariates
> NOTA: in realtà questa figure l’ho tolta, alla prof non piaceva, e in effetti non trasmetteva molte informazioni
> **Again, this was performed in order to remove any trend or bias on the covariates and to equalize their contribution at each time instant.**

Nota 8

- [x] Dalle Figure 3.17 e 3.18 non si capisce molto come va la MCMC, sono troppe time series sovrapposte. Io farei vedere qualche trace plot per mostrare che la catena converge (quindi dovrebbero essere i soliti bruchi pelosi, per intenderci) e poi manterrei la stessa struttura delle Figure 3.17 e 3.18, ma al posto di plottare tutte le time series plotterei i credible intervals (con la media indicata da una linea più scura), magari collegandoli tra i vari istanti temporali.

Qui ci sta mi piace molto l'idea di questi nuovi plot. L'ho inserita ma non so se il plot è ancora troppo "pieno"; è di sicuro ancora denso ma ora occupa una pagina tutta sua, quindi questo dovrebbe dargli più aria. Avevo pensato ad un layout con 4 righe e 2 colonne, 2 righe sopra per due trace plot e le altre 3 per i 6 credible intervals. Ma mettendo due soli trace plot magari veniva il dubbio di quali mettere, perché hai messo proprio quelli, cosa nascondono gli altri, ecc.

La riporto anche qui comunque.



Nota 9

- [x] Le Figure 3.26, 3.27 e 3.28 hanno moltissime informazioni, quindi per un lettore non è facilissimo captare subito le differenze tra le varie figure. Suggerirei, nella discussione alla fine della Sezione 3.2.2, di mettere qualche esempio riferito alle figure dove si vede subito la differenza tra i clustering, così un lettore sa subito dove guardare.

Fatto. Ho inserito nel testo un'indicazione di in quali istanti temporali si possa osservare meglio l'effetto dato dalle covariate. L'ho scritto nel testo ma anche nella caption dell'immagine. Magari poi la approfondisco quando rifinisco il testo.

The effects of the covariates contributions can be seen more clearly at time instants 8, 9, and 12.

Nota 10

- [ ] Aggiungerei qualche chiacchiera su cosa vuol dire avere le covariate nella likelihood rispetto al clustering: avere le covariate solo nella likelihood vuol dire considerare solo l'effetto lineare della covariata, avere le covariate solo nel clustering vuol dire considerare tutti gli effetti della covariata nel clustering (quindi, tanti cluster molto specifici), avere le covariate sia nella likelihood che nel clustering vuol dire considerare gli effetti lineari nella likelihood e solo gli effetti nonlineari nel clustering (quindi, meno cluster). Puoi trovare un piccolo paragrafo che ne parla nella sezione 3.2 di Quintana, F. A., Müller, P., & Papoila, A. L. (2015). Cluster-Specific Variable Selection for Product Partition Models. Scandinavian Journal of Statistics, 42(4), 1065–1077. https://doi.org/10.1111/sjos.12151 (https://doi.org/10.1111/sjos.12151).

Questa magari la aggiungo nella sezione in cui facciamo il test con l'intera time series a NA. Pensavo di fare quindi queste sezioni nella sezione "effects of the covariates" del capitolo Testing:

1. covaraites in the likelihood
2. covariates in the prior (anziché in the clustering che è meno corretto, segnalato dalla prof)
3. inference in/on a new location (in cui fare il test con le missing time series)

e quindi scrivere nella 3 quei commenti sull'interazione delle covariate.

Nota 11

- [x] Nella sezione 3.4, aggiungerei qualche commento testuale per sottolineare il miglioramento che si vede dalle immagini (visto che è uno dei punti forti della tesi). Per esempio, potresti dire che con T=50 il JDRPM va meglio del modello C anche se utilizza 5 covariate sia nella likelihood che nel clustering. Inoltre, ricorderei i valori di n, T e p dell'applicazione che hai presentato prima, per mostrare che in un'applicazione reale la tua implementazione comporta un miglioramento delle prestazioni.

Aggiunto del testo che marca questo miglioramento. Devo riformularlo e convertirlo in un inglese più corretto e professionale ma questa è l'idea.

> Figure \ref{fig: scaling target space covariates} shows how fits including covariates saw a drop of performance, which was inevitable due to the additional work to perform, but they still remain at a good performance level. Indeed, Figure \ref{fig: summary performance scaling} will point out how some of the fits with all information levels in Julia were actually faster than the basic fits in C**; which is quite surprising given the additional complexity and computations introduced by the covariates.**
>
> **Also in the noisier environment in which the real-application fits of Section \ref{Target variable plus space} were performed, where we did not try, as instead we did in this dedicated section, to devote all the hardware resources to the models fitting, a considerable speedup was observed. In that case, in which the dataset consisted of $n=105$ units and $T=12$ time instants, the CDRPM fit with target plus space values took 1 hour and 38 minutes, while both the JDRPM fits, with equivalent setup and with covariates in the clustering, exhibited faster execution times. Remarkably, the equivalent JDRPM fit more than halved the execution time; while the fit including covariates showed the expected speedup reported by Figure \ref{fig: summary performance scaling}, if we look at the closest corresponding case of $n=100$ and $T=10$. This is definitely a further proof of the effectiveness of the new implementation.**

Nota 12

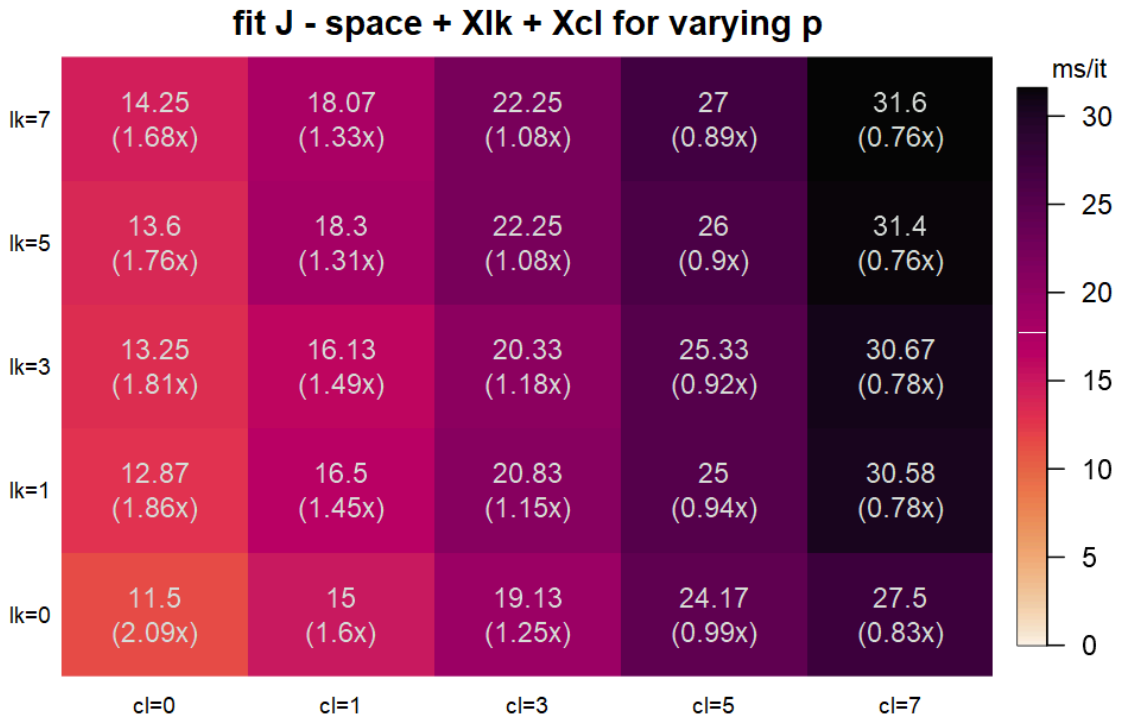
- [x] Farei anche uno studio in cui, fissati n e T a valori ragionevoli (per esempio, 50 e 50), verifichi il tempo computazionale al crescere di p. In questo modo, dovresti trovare il numero di covariate tale per cui le tue performance arrivano alle performance di CDRPM (senza covariate).

Fatto. È una singola matrice in cui ci sono fit julia con spazio + xcl + xlk, con p che variano da 0 (cioè assenti) a ECC. Pensavo fosse un modo sensato e comodo per vedere l'effetto dell'aggiunta di quante covariate e in quali zone (se in un livello, nell'altro, o entrambi).

Il testo è per ora uscito così: (sempre un po' provvisorio comunque, devo sistemarlo ancora per la versione finale)

Figure \ref{fig: scaling target space covariates} shows how fits including covariates saw a drop of performance, which was inevitable due to the additional work to perform, but they still remain at a good performance level. Indeed, Figure \ref{fig: summary performance scaling} will point out how some of the fits with all information levels in Julia were actually faster than the basic fits in C; which is quite surprising given the additional complexity and computations introduced by the covariates.

From the encouraging results of Figure \ref{fig: summary performance scaling}, we also investigated at which model complexity the JDRPM implementation would tie the execution times proposed by CDRPM. To do so, we selected a possibly regular-sized dataset, of $n=50$ and $T=50$, and we took as reference the execution time per iteration exhibited by the CDPRM fit in the case of target plus space values. Figure \ref{fig: scaling target space} retrieves this value to be 24 ms/it. Then, we evaluated the performance of JDRPM fits including covariates, at likelihood and/or clustering levels, letting now p be varying, to see at which degree of complexity the new implementation match the original one. Figure \ref{fig: test varying p} summarizes the result of this analysis. It appears that until we include $p_\text{cl}=5$ covariates in the clustering, we can expect the JDRPM implementation to be faster than the simple spatially-informed fit on CDRPM. This means that in the same time in which the CDRPM performed a spatially-informed fit, JDRPM can perform the same fit plus up to five covariates in the clustering and a possibly indifferent number of covariates in the likelihood, since the increase of p_lk did not highlight any significance drop in the performances.



As in the previous tests of this section, this analysis was performed trying to devote all the computational resources to the fitting task; therefore the results should be accurate. As a further proof, the comparison performed in Section \ref{Covariates in the clustering} and summarized in Table \ref{tab: fits metrics space all with also Xcl} showed a fit with JDRPM plus three covariates in the clustering to take 1h20m. Corresponding to this case ($p_\text{lk}=0$, $p_\text{cl}=3$), Figure \ref{fig: test varying p} suggests a speedup factor of 1.25x; and indeed this would imply an expected time of 1h40m in the CDRPM fit, which is practically matching the measured one of 1h38m.

Nota 13

- [x] Riguardo l'appendice C, secondo me puoi lasciarla come ultima cosa da fare se avanza tempo. La tua tesi non è applicativa, l'applicazione è utile soltanto per far vedere che il modello è sensato. Perciò, una analisi interpretativa dei risultati mi sembra qualcosa di molto marginale.

Ok va bene, vediamo se avanza tempo nei prossimi giorni.

Domande:

1. Nella appendice computational details quindi può aver senso lasciare il codice julia? perché magari se uno vuole leggerlo se lo scarica da github. Nel senso, non so se sia una pratica comune nelle tesi inserire il codice che implementa un qualche modello. Però forse ci sta per quello che ho scritto nei paragrafi prima, e per mostrare un po' tutto il lavoro di programmazione

Appendix B

Computational details

B.1 Fitting algorithm code

We report here part of the code of the `MCMC_fit` function which implements the updated DRPM model described in this work. We include it to let the readers appreciate the ease, clarity, and even elegance of the Julia language in translating the mathematical formulation into code. All this with the the hope for Julia to become the new natural choice in the statistical, or in general scientific, computing field, giving to it a refreshing approach.

Together with what said in Chapter 2, we also note that the productivity offered by the Julia language is not only obtained through the vast land of packages and documentation, but even through an online active forum[1], where I wrote myself some questions (and received answers) during the development of the thesis. Finally, at https://github.com/federicomor/Tesi/tree/main/src/JDRPM there are available all the codes behind this JDRPM update.

Listing 3: Julia code which implements the fitting model algorithm. We report here just the "functional" part, i.e. not all the setup lines regarding the function definition, variable preallocations, input arguments checks, etc.

```
############ start MCMC algorithm #############
println(replace(string(now()),"T" => " ")[1:end-4])
println("Starting MCMC algorithm")

t_start = now()
progresso = Progress(round(Int64(draws)),
      showspeed=true,
      output=stdout, # default is stderr, which turns out in orange color on R
      dt=1, # every how many seconds update the feedback
      barlen=0 # no progress bar
      )

@inbounds for i in 1:draws
   ############ sample the missing values #############
   # from the "update rho" section onwards also the Y[j,t] will be needed (to
   ↪  compute weights, update laws, etc)
   # so we need now to simulate the values for the data which are missing (from
   ↪  their full conditional)
```

[1] https://discourse.julialang.org/

2. le altre le ho già messe nelle risposte alle sue note, tra cui per esempio le consulenze sui plot