Optimizing the JSM Program

Luca Frigau, University of Cagliari, frigau@unica.it
Qiuyi Wu, University of Rochester, Qiuyi_Wu@URMC.Rochester.edu
David Banks, Duke University, dlbanks@duke.edu

August 25, 2021

Abstract

Sometimes the Joint Statistical Meetings (JSM) is frustrating to attend, because multiple sessions on the same topic are scheduled at the same time. This paper uses seeded Latent Dirichlet Allocation and a scheduling optimization algorithm to very significantly reduce overlapping content in the original schedule for the 2020 JSM program. Specifically, a measure based on total variation distance that ranges from 0 (random scheduling) to 1 (no overlapping content) finds that the original schedule had a score of 0.058, whereas our proposed schedule achieved a score of 0.371. This is a huge improvement that would (1) increase participant satisfaction as measured by the post-JSM satisfaction survey, and (2) save the American Statistical Association significant money by obviating the need for the traditional in-person meeting of the 47 program chairs and other organizers. The methodology developed in this work immediately applies to future JSMs and is easily modified to improve scheduling for any other scientific conference that has parallel sessions.

Keywords: Latent Dirichlet Allocation; Topic Modeling; Greedy Algorithms; Optimal Scheduling

1 Introduction

Each year, the Joint Statistical Meetings (JSM) brings together several thousand statisticians for a significant conference. Organizing its program is always a challenge. After every meeting, the American Statistical Association (ASA) administers a participant satisfaction survey to identify ways to improve the management and structure of the conference, and the largest and most consistent complaint is about subject matter conflicts in concurrent sessions. At the last in-person JSM in 2019, 106 out of 502 respondents to the satisfaction survey raised this issue, and in 2018, it was 88 out of 426 respondents.

To reduce subject conflicts, once a year the ASA flies in all the members of the Program Committee to its headquarters in Alexandria, Virginia, in order to finalize the program schedule and to collect contributed abstracts into themed sessions. In 2020, this meeting involved 47 organizers and three ASA staff, and it lasted 1.75 days. Its key task is to minimize overlapping content in the same time band among the invited, topic contributed, and contributed sessions. This travel and lodging is a significant financial expense for the ASA, and the time required is a significant burden upon the Program Committee.

The JSM program is set by the Program Committee. It consists of the program chairs of each section, representatives from sister societies participating in the JSM, three overall program chairs (past, present, and future), and three associate chairs. Its size varies between even and odd years, according to whether the Institute of Mathematical Statistics holds its annual meeting at the JSM.

In 2020, the Program Committee developed a schedule that improved over random assignment by 5.8% (with respect minimizing overlapping content). The method described in this paper improves the same program by 37.1%. As discussed later, any additional improvement would be negligible.

Some large conferences try to minimize content overlap by creating tracks; e.g., in 2021, INFORMS has tracks on Healthcare, Analytics Leadership, Decision Analysis, and so forth. The JSM has resisted that rigidity, but at the price of expensive and

labor-intensive committee meetings and imperfect scheduling. This paper uses a flexible alternative strategy for organizing the JSM that uses Latent Dirichlet Allocation (LDA) and optimal scheduling to minimize content conflicts among parallel sessions.

There are seven categories of sessions at the JSM: plenaries, invited sessions, topic contributed sessions, contributed sessions, introductory overview lectures, late-breaking sessions, and memorial sessions. (There are also breakfast and lunch roundtables, poster sessions and speed sessions, but we do not address those in this paper.) In 2020, we planned to have up to 44 rooms available for parallel sessions, with three 110 minute time bands on Monday, Tuesday and Wednesday and two 110 minute time bands on Sunday and Thursday.

Our data on the 2020 JSM program consist of the title, keywords, and abstract text (as well as speaker name and affiliation, which we do not use) for each talk in each session. We want to use that information to ensure that, e.g., there are not six sessions on clinical trials which are all scheduled for 8:30 a.m. on Tuesday morning. To boost signal, we enrich the data by representing the title of each talk and the associated key words three times.

Our approach is to use an extension of LDA to automatically identify broad topics (e.g., environmental statistics, time series analysis, survey methodology, and so forth). A session typically participates in only a handful of those topics. LDA defines a topic as a distribution over the words in a dictionary; if a topic puts significant weight on "non-response" and "questionnaire" and "strata", then it is easy for a human to recognize that topic as survey methodology. (This description of LDA is an oversimplification—see Section 3 for details.)

LDA uses Markov chain Monte Carlo (MCMC) to estimate the extent to which each session participates in each of the topics; e.g., a session might be 50% about clinical trials, 40% about survey methodology, and 10% about maximum likelihood estimation. Presumably, that session would have talks describing how participants in a clinical trial might be surveyed to discover such things as unforeseen side-effects. Those topic percentages summarize the content of each session. Our goal is to minimize the overlap in topics within the same time slot.

Given those topic percentages, we then use a scheduling algorithm to assign sessions to time bands such that the amount of conflict among parallel sessions is minimized. Although optimal scheduling is combinatorially hard (it is a form of the knapsack problem), good heuristic search can achieve schedules that significantly improve over the actual scheduling at the JSM. Section 4 describes the algorithm and compares it to the schedule planned for the 2020 program before the pandemic forced JSM to become virtual.

Section 2 describes the preparation and cleaning of the data. Section 3 explains the LDA implementation, and how the number of topics was determined. Section 4 defines the scheduling algorithm, the result it obtains, and shows its significant improvement over the 2020 program in terms of total conflict. Section 5 is a brief discussion, and urges the American Statistical Association to use this method when scheduling future JSMs (at least to draft a first-pass schedule, which could be modified as the Program Committee deems necessary).

2 Data Preparation

The first step was to remove stop words. Stop words contain little information about the topic. Examples of stop words are "a", "of", "the", and so forth. There are many lists of stop words; they are all similar, and we merged the lists *Lingua::StopWords* (https://metacpan.org/pod/Lingua::StopWords) and the *Stopwords ISO* (Benoit et al., 2019).

The second step was to stem the words. This identifies different forms of a word by a single token. There is little information lost by consolidating "estimate", "estimates", "estimation" and "estimating". This reduces the vocabulary, which speeds the computation. We used the Snowball stemmer (Porter, 2001); this if fully automated, and requires no domain knowledge. Also in this step we removed any token that appeared fewer than 5 times, to speed computation, and those that appeared in only a single session, since these provide no information relevant to minimizing schedule conflicts.

The third step was n-gramming. This procedure looks for sequences of tokens that co-occur improbably often. For example, let p_1 be the frequency in the corpus of abstracts of the token for "maximum" and let p_2 be the frequency of the token for "likelihood". We anticipate that the empirical frequency of the sequence "maximum likelihood" will be much greater than $p_1 p_2$, the frequency expected under independence. This is because the words are not independent—the phrase has a specific technical meaning, so the concept is best represented by a single token rather than by two tokens. So whenever the abstracts mention "maximum likelihood", that phrase is replaced by its own unique token.

The threshold for an n-gram occurring improbably often is that (1) there are at least 5 co-occurences and (2) the standard binomial test of whether the observed proportion of co-occurences exceeds that expected under independence has significance probability less than 0.005. The reason for this stringent threshold is that the independence model is not well satisfied in English writing; there are many common n-grams which are not statistically relevant. Our threshold encourages the selection of n-grams that are technical phrases. For that same reason, we do not seek n-grams of length greater than five tokens; "uniformly most powerful unbiased test" seems sufficiently specific.

There are many more sophisticated ways of identifying n-grams, based on moving windows that have some prespecified number of tokens in common and other techniques (Sidorov, 2019). But our results indicate that this simpler choice performs well.

The fourth step is unusual, but useful in this application. There are many words that are not stop words, but which do not convey information about the scientific content of a session (e.g., "therefore", "however", "follows"). Similarly, there are many common phrases that are unhelpful in identifying content (e.g., "responsible leadership" or "these figures show"). We want to eliminate these, to reduce computation and increase the signal-to-noise ratio in the data.

Our approach to doing this is to take a non-statistical text, apply steps one to four as described above, and then remove all tokens from the JSM corpus that also

appear in the non-statistical text (except for tokens included in the seeded words list described in Section 3). The text we used was the Trayvon Martin corpus, a collection of political blog posts from 2012 (Soriano et al., 2013), and we chose it because it is almost entirely non-statistical and because one of us had worked with it before and it had already been cleaned, stemmed and n-grammed. This corpus uses the word "therefore" but not the word "autoregressive" and thus we remove the uninformative tokens. Similarly, the corpus uses phases such as "everyone agrees" which appear in the JSM corpus but which do not help identify topics. A manual review of the terms removed in this step found no problematic deletions—the only possible exception was the bigram tokenizing "statistical evidence", but we did not feel this was particularly topic specific.

We also manually removed the tokens corresponding to "statistician" and "statist". These are essentially useless in discriminating among topics at the JSM (but we did not remove n-grams that contain that term, such as "statist_significan" and "statist_scien").

These preprocessing steps are always needed in text mining. Almost two decades of experience with the tools used in this paper indicate that the results are insensitive to the very standard choices we have made, and this belief is supported by the sensitivity analysis described in the next Section 3.

The first step reduced the number of words to 21,813 tokens. The second step and third steps decreased the number of tokens to 17,471, of which 14,017 are n-grams. Often the tokenization will automatically create near synonyms—one can have distinct tokens for "time series" and "time series analysis". The fourth step, using the Trayvon Martin corpus, reduced the number of tokens to 15,755. The computing time needed for the data preparation was 285 seconds on a MacBook Pro, 2.8 GHz Intel Core i7, 16 GB 1600 MHz DDR3.

3 Seeded LDA

Latent Dirichlet Allocation (LDA) was developed by Blei et al. (2003) and has become a popular tool. Its generative model assumes that the tokens used in a document in the corpus are drawn from K topics, where each topic is a probability distribution over the available tokens. With probability π_{ik}, independently, the ith document draws from the kth topic, and the number of draws equals the number of tokens in the ith document. In applications, MCMC reverses the generative process, estimating the distributions corresponding the topics and the probabilities $\pi_{i1}, \ldots, \pi_{iK}$ for all documents. In our application, a document corresponds to a JSM session.

More precisely, vanilla LDA ostensibly starts with a Poisson prior over the number of topics that appear in the corpus, but in practice most researchers try several reasonable values and then fix the number of topics to equal a value that provides interpretable topics. This approach is essentially what was done in this paper.

The real specification of the generative model starts by assuming a Dirichlet distribution with parameter α over the N retained tokens, and making K draws from that distribution. Denote these by ϕ_k. Those draws determine the probabilities that each of the K topics places upon the tokens, so these vectors have length N. Thus if ϕ_k puts a lot of weight on tokens such as "calculus" and "homotopy" and "combinatorics", then it is easy for a human to interpret topic k as "Mathematics". But if the draw put a lot of weight on "magnetism", "gravity" and "Planck's constant", then it would be interpreted as "Physics". Clearly, in this formulation, all topics will put a little weight on each token, so the interpretation is driven by the high-probability tokens.

Next, for the ith document, one draws from a second Dirichlet with parameter β. Denote this draw by θ_i. This θ_i has length K and determines the extent to which document i participates in each of the K topics. For example, if that draw put 30% of the weight on "Mathematics" and 70% on "Physics", then a human would interpret the ith document as being about mathematical physics. Again, this framework implies that all documents will have some participation in every topic, but one hopes that a

small number of dominant topics will emerge for each document.

Finally, to generate each of the tokens in the ith document, one first draws z_j from a one-trial multinomial with with parameter θ_i, in order to pick the topic that generates that token—suppose it is topic k. Then one draws from a one-trial multinomial with parameter ϕ_k to determine which token within that topic is chosen for the document. This creates a "bag-of-words" representation for the ith document (i.e., token order does not matter, but all the tokens used reflect the weighted participation by the different topics).

Using this generative model, MCMC reverse engineers the process to obtain estimates of the K topic distributions and the extent to which each document participates in each topic. As usual, convergence is assessed through a traceplot. Also as usual, assessment of fit is based upon the interpretability of the discovered topics and quantified via perplexity, as in Blei et al. (2003).

There are many variants of LDA, but this paper hews as close to vanilla LDA as possible, in order to draw upon the two decades of experience that researchers have with such modeling. There are a number of tuning parameters used in this analysis, and we do not want to stray too far from conventional wisdom.

Seeded LDA is similar to LDA, except that there is the added constraint that many topics are forced to put zero weight on specific tokens (Jagarlamudi et al., 2012). In our example, if we wanted to force one topic to contain terms related to, say, astrostatistics, then we would require that all but one of the topics have zero weight on the token corresponding to "astrostatistics". The one topic that allowed positive weight for that token would also accumulate related tokens such as those corresponding the "extra-solar planets" and "red-shift". Sessions that mention "astrostatistics" would also be likely to use those additional tokens, and thus the generative model would place them all in the same topic and give them relatively large weights. Non-seed tokens may have positive probability in more than one topic, but seed tokens have positive weight in only one topic.

Seeding forces topic distributions to pertain to prespecified subjects. Instead of automatically discovering topics, as in LDA, we use seeded LDA to find which tokens

have high probability within partially prespecified topics that correspond to distinct subfields of statistics at the JSM.

Our application is to the JSM program, whose structure is governed by the Joint Agreement among the four founding societies: the ASA, the Institute of Mathematical Statistics, the International Biometric Society, and the Statistical Society of Canada. In even years, there are 181 invited sessions; in odd years, the Institute of Mathematical Statistics holds its annual meeting at the JSM, and then there are 209 invited sessions. Our work used the governance structures of the founding societies to guide the seeding. In particular, we created topic seeds for nearly every section of the ASA.

For the 2020 meeting, invited sessions could only be sponsored by ASA sections, interest groups, journals (for session slots allocated on a rotating basis among the ASA journals, except that the *Journal of the American Statistical Association* always has exactly two), the Leadership Support Council, the Council of Chapters, ASA committees, certain named awards, and other professional societies (with varying degrees of privilege, based on the Joint Agreement).

In order to seed LDA we identified words that are reasonably specific to topics we want to define. We looked at the stemmed words' frequencies and assigned fairly specific but relatively common tokens to topics. This process excluded extremely frequent stemmed words, such as *bayesian*, *infer*, *asymptot*, *gaussian* and *causal*, because they were insufficiently specific, and naturally appear in many different topics. But we did use them when they appeared in n-grams that corresponded to more narrow topics, such as *bayesian_nonparametr*, *asymptot_distribut*, *gaussian_process* and *causal_infer*. We tried to assign approximately equal numbers of tokens to each topic, but there was variation; e.g., we used a relatively large number of stemmed words for the topic *Genomics/Genetics*, since all of them seemed informative. The list of stemmed words used to seed the topics is shown in Table 3.

Additionally, we added ten unseeded topics, to account for areas we may have overlooked. The choice of ten was a judgment call. At a fine scale, Soriano et al. (2013) found five clearly interpretable topics within the Trayvon Martin corpus. But Henry et al. (2019), in an analysis of all 2012 political blogs, which includes that

corpus, chose to focus on coarse structure, finding 22 clearly interpretable topics. We felt that we probably had not overlooked too many topics, and this guess was supported by the fact that seven of the unseeded clusters were reasonably coherent, but three were not.

This was not problematic, but variational inference can be used for larger problems. The traceplots strongly corroborated convergence, and in LDA, convergence has generally not been an issue unless the texts are very short, such as tweets (Kim and Shim, 2014).

The use of unseeded topics allows the data to speak for itself. For example, in this framework, it is unclear where tokens for n-grams corresponding to "binomial distribution" or "central limit theorem" or "multiple regression" would go; they might appear in multiple topics, or they might group together in an unseeded topic that could be interpreted as, say, elementary statistics. So we performed LDA with $K = 51$ topics: 41 seeded and 10 unseeded.

In order to assess the quality of the resulting topic estimates, we calculated the distinctivity of the words found in that topic. Here, the distinctivity of the jth token for the kth topic is the posterior probability of the kth topic given that the jth token appears in that session's text, for a uniform prior over the topics. Highly distinctive tokens are specific to a single topic; tokens with smaller posterior probabilities are associated with multiple topics. Table 3 lists the five most distinctive tokens for each of the seeded topics, and their associated posterior probabilities. We have rounded the distictivities to the first decimal place; additional decimal places would be noisy and convey little information. Overall, tokens are sensibly matched to their seeded topics.

Seeded Topic	Stemmed word	Seeded Topic	Stemmed word
Agriculture	agricultur	Graphics	data_visual
Agriculture	isoton_regress	Graphics	graphic
Agriculture	nass	Graphics	graphic_model
Agriculture	quantil_regress	Graphics	visual
ASA	cyber	Health Policy	health_care
ASA	dod	Health Policy	influenza
ASA	secur	Health Policy	opioid
Astrostatistics	astronom	Health Policy	protocol
Astrostatistics	astronomi	Health Policy	public_health
Astrostatistics	astrophys	High dimension	dimens_reduct
Astrostatistics	astrostatist	High dimension	dimension_reduct
Asymptotics	asymptot_distribut	High dimension	model_high_dimension
Asymptotics	asymptot_normal	Imaging	convolut_neural_network
Asymptotics	asymptot_properti	Imaging	imag_data
Bayesian Computing1	bayesian_comput	Imaging	satellit_imageri
Bayesian Computing1	hamiltonian	Inference	infer_procedur
Bayesian Computing1	scalabl_estim_bayesian	Inference	inferenti
Bayesian Computing2	baroreceptor	Inference	statist_infer
Bayesian Computing2	delay_treatment	Learning and Data Science	classif_tree
Bayesian Non-parametric	bayesian_nonparametr	Learning and Data Science	lasso
Bayesian Non-parametric	dirichlet_process	Learning and Data Science	random_forest
Bayesian Non-parametric	nonparametr_bayesian	Learning and Data Science	semi_supervis_learn
Biopharmaceutica	biomark	Lifetime Data	longitudin_data
Biopharmaceutica	biomed	Lifetime Data	surviv_analysi
Biopharmaceutica	microbiom	Lifetime Data	surviv_data
Biopharmaceutica	microbiom_data	Longitudinal Analysis	longitudin_measur
Biopharmaceutica	placebo	Longitudinal Analysis	longitudin_non_gaussian
Business/Economic	advertis	Longitudinal Analysis	longitudin_studi
Business/Economic	econom_growth	Medical imaging	brain_imag
Business/Economic	macroeconom	Medical imaging	fmri
Business/Economic	market	Medical imaging	medic_imag
Causal Inference	causal_infer	Medical imaging	neuroimag
Causal Inference	causal_mediat	Mental Health	alzheim
Causal Inference	counterfactu	Mental Health	dementia
Causal Inference	estim_causal	Mental Health	late_onset_alzheim
Climate	climat_chang	Mental Health	mental_health
Climate	climat_model	Missing Data	imput
Climate	climatolog	Missing Data	mi
Clinical Trials	adapt_clinic	Missing Data	multipl_imput
Clinical Trials	dose	Network Analysis	network_analysi
Clinical Trials	dose_find	Network Analysis	network_structur
Clinical Trials	oncolog	Network Analysis	node
Clinical Trials	phase_trial	Network Analysis	social_network
Clinical Trials	trial_design	Network Analysis	topolog
Data Science Education	ap	Nonparametrics	nonparametr_estim
Data Science Education	data_scienc_educ	Nonparametrics	permut
Data Science Education	statist_educ	Risk Analysis	compet_risk
Data Science Education	undergradu	Risk Analysis	cox_model
Deep learning	deep_learn	Risk Analysis	cox_proport
Deep learning	deep_neural_network	Risk Analysis	risk_factor
Design of Experiments	adapt_design	Risk Analysis	risk_predict
Design of Experiments	experiment_design	Sport	basebal
Design of Experiments	foldov	Sport	basketbal
Design of Experiments	foldov_techniqu	Sport	chess
Design of Experiments	sampl_design	Sport	footbal
Design of Experiments	split_plot_design	Sport	ice_hockey
Design of Experiments	studi_design	Sport	sport
Environment	air_pollut	Statistical Computing	comput_effici
Environment	ecolog	Statistical Computing	python
Environment	environment_exposur	Statistical Computing	quantum
Epidemiology	epidem	Statistical Computing	tensor
Epidemiology	epidemiolog	Stochastic Process	gaussian_process
Epidemiology	hazard	Stochastic Process	process_prior
Epidemiology	rare_diseas	Stochastic Process	stochast_process
Genomics/Genetics	cell_type	Survey Research	complex_survey
Genomics/Genetics	dna_methyl	Survey Research	survey_data
Genomics/Genetics	gene_express	Survey Research	survey_design
Genomics/Genetics	genom_data	Survey Research	survey_sampl
Genomics/Genetics	genome_wid_associ	Time Series	smooth
Genomics/Genetics	genotyp	Time Series	time_seri
Genomics/Genetics	herit	Topic Modelling	latent_dirichlet_alloc
Genomics/Genetics	rna	Topic Modelling	text_min
Genomics/Genetics	singl_cell	Topic Modelling	topic_model
Genomics/Genetics	variant	Transportation	driver
Government	administr_data	Transportation	exposur_traffic_rel
Government	administr_record	Transportation	traffic
Government	census	Transportation	traffic_rel
Government	mentor	Transportation	transport
Government	nces	Uncertainty Quantification	emul
		Uncertainty Quantification	uncertainti_quantif

Table 1: The list of 159 stemmed words used in seeded LDA grouped by 41 seeding topics. Stemmed words divided by "_" indicate n-grams.

Agriculture	prob	American Defense	prob	Astrostatistics	prob	Asymptotics	prob	Bayesian Computing1	prob	Bayesian Computing2	prob
isoton_regress	0.6	dod	0.9	astronom	1.0	asymptot_properti	0.9	hamiltonian	1.0	baroreceptor	0.8
hot_flash	0.6	cyber	0.9	astrostatist	1.0	asymptot_distribut	0.9	bayesian_comput	1.0	delay_treatment	0.8
nass	0.5	secur	0.7	astrophys	1.0	asymptot_normal	0.9	scalabl_estim_bayesian	0.9	ecowa	0.6
allergen_immunotherapi	0.5	deidentif	0.6	astronomi	0.9	quantil	0.8	hamiltonian_mont_carlo	0.7	fish_effort	0.6
max_combo_test	0.5	growclust	0.5	stellar	0.8	quantile_regress	0.8	hamiltonian_mont	0.7	water_violat	0.5
Bayesian Non-parametric	prob	Biopharmaceutica	prob	Business/Economic	prob	Causal Inference	prob	Climate	prob	Clinical Trials	prob
nonparametr_bayesian	1.0	microbiom	1.0	advertis	1.0	counterfactu	1.0	climatolog	0.9	oncolog	1.0
dirichlet_process	0.9	microbiom_data	1.0	econom_growth	0.9	causal_infer	1.0	climat_chang	0.9	adapt_clinic	1.0
mri	0.8	microbi	0.9	rwd	0.9	causal_mediat	0.9	expo	0.8	dose_find	1.0
magnet_reson	0.8	phylogenet	0.9	rwe	0.9	unmeasur_confound	0.9	data_challeng_expo	0.6	phase_trial	1.0
voxel	0.8	biomed	0.9	market	0.9	unmeasur	0.9	challeng_expo	0.6	trial_design	1.0
Data Science Education	prob	Deep learning	prob	Design of Experiments	prob	Environment	prob	Epidemiology	prob	Genomics/Genetics	prob
data_scienc_educ	1.0	deep_learn	1.0	experiment_design	1.0	environment_exposur	1.0	rare_diseas	1.0	cell_type	1.0
ap	1.0	deep_neural_network	1.0	split_plot_design	1.0	ehr	0.9	epidemiolog	0.9	rna	1.0
statist_educ	1.0	neural	0.9	foldov	0.9	air_pollut	0.9	trial_rare_diseas	0.7	genom_data	1.0
undergradu	1.0	neural_network	0.9	foldov_techniqu	0.9	wearabl	0.9	epidem	0.7	singl_cell	1.0
data_scienc	1.0	machine_learn	0.9	d_optim	0.7	ecolog	0.9	gene_therapi	0.6	variant	1.0
Government	prob	Graphics	prob	Health Policy	prob	Health Policy	prob	Imaging	prob	Inference	prob
mentor	1.0	graphic_model	1.0	opioid	1.0	dimens_reduct	1.0	convolut_neural_network	1.0	infer_procedur	0.6
census	1.0	data_visual	1.0	influenza	1.0	model_high_dimension	0.9	satellit_imageri	0.9	influenc_matrix	0.6
administr_record	1.0	graphic	0.9	health_care	0.9	dimension_reduct	0.9	imag_data	0.8	inferenti	0.5
administr_data	0.9	visual	0.9	diagnost_test	0.8	suffici_dimens_reduct	0.9	empir_bay	0.8	slice_wasserstein	0.5
undercount	0.9	patient_specif	0.8	life_expect	0.8	suffici_dimens	0.9	epistasi	0.7	spectrum_estim	0.5
Learning and Data Science	prob	Lifetime Data	prob	Longitudinal Analysis	prob	Medical imaging	prob	Mental Health	prob	Missing Data	prob
classif_tree	1.0	surviv_data	1.0	longitudin_measur	0.8	medic_imag	0.6	mental_health	0.9	mi	1.0
semi_supervis_learn	1.0	surviv_analysi	0.9	longitudin_non_gaussian	0.8	dili	0.6	late_onset_alzheim	0.9	multipl_imput	1.0
random_forest	0.9	survival_analysi	0.9	sepsi	0.6	fractal_dimens	0.6	alzheim	0.9	imput	0.9
biomark_predict	0.7	cluster_size	0.9	spatial_repel	0.5	fractal	0.5	dementia	0.9	missing_data	0.9
diabet_kidney_diseas	0.7	inform_cluster	0.8	sot	0.5	multiplex	0.5	s_diseas	0.8	miss	0.9
Network Analysis	prob	Nonparametrics	prob	Risk Analysis	prob	Sport	prob	Statistical Computing	prob	Stochastic Process	prob
network_analysi	1.0	nonparametr_estim	1.0	cox_model	1.0	footbal	0.9	quantum	1.0	process_prior	0.9
topolog	1.0	permut	0.9	risk_predict	0.9	basketbal	0.9	tensor	1.0	stochast_process	0.9
network_structur	1.0	benefit_risk_assess	0.8	cox_proport	0.9	chess	0.9	python	0.9	spatio_tempor	0.9
social_network	0.9	permut_test	0.8	risk_factor	0.7	basebal	0.9	gradient_descent	0.8	downscal	0.9
node	0.9	primari_endpoint _secondari	0.7	compet_risk	0.6	ice_hockey	0.9	comput_effici	0.8	spatial_model	0.9
Survey Research	prob	Time Series	prob	Topic Modelling	prob	Transportation	prob	Uncertainty Quantification	prob		
survey_design	1.0	time_seri	0.9	latent_dirichlet_alloc	0.8	driver	1.0	refer_region	0.6		
complex_survey	1.0	nowcast	0.9	text_min	0.7	transport	1.0	drug_reposit	0.6		
survey_sampl	0.9	autoregress	0.9	topic_model	0.6	traffic	0.9	reposit	0.5		
rake	0.9	autoregress_model	0.9	analyt_curriculum	0.6	exposur_traffic_rel	0.9	toxic_burden	0.5		
acreag	0.8	seri_data	0.9	teacher_stud	0.5	traffic_rel	0.9	prior_select	0.5		

Table 2: The list of 5 most distinctive stemmed words in the 41 seeded topics. Stemmed words divided by "_" indicate n-grams.

Similarly, Table 3 shows the five most distinctive words for the ten unseeded topics, and their associated posterior probabilities. Unseeded topics 1, 3, and 10 seem difficult to identify, and probably have absorbed noise rather than signal. (In topic 3, dtr is the acronym for dynamic treatment regime and nri is the acronym for National Resources Inventory.) Unseeded topic 2 relates to the environment, topic 4 concerns cluster analysis, topic 5 is about additive regression trees, 6 is about MCMC, 7 is about false discovery rates, and 8 is about regression.

Unseeded topic 1	prob	Unseeded topic 2	prob	Unseeded topic 3	prob	Unseeded topic 4	prob	Unseeded topic 5	prob
misbehavior	0.7	spis	0.6	nri	0.8	trio	0.8	bart	0.9
instructor_misbehavior	0.6	metal_exposur	0.6	dtr	0.8	prompt	0.8	addit_regress_tree	0.8
rotavirus	0.6	ebolavirus	0.6	underdispers	0.7	model_bas_cluster	0.8	bayesian_addit_regress	0.8
tax_audit	0.5	built_environ	0.5	formula	0.7	latent_class_model	0.7	bayesian_addit_regress_tree	0.8
random_quantil	0.5	risk_differ	0.5	repeat_measur	0.7	parent_child	0.7	bayesian_addit	0.8
Unseeded topic 6	**prob**	**Unseeded topic 7**	**prob**	**Unseeded topic 8**	**prob**	**Unseeded topic 9**	**prob**	**Unseeded topic 10**	**prob**
bayesian_infer	0.9	knockoff	0.9	linear	0.9	haplogroup	0.6	rasch_tree	0.6
posterior	0.9	fals_discoveri	0.9	general_linear_model	0.9	power_prior_distribut	0.6	demand_elast	0.5
variational_infer	0.9	fals_discoveri_rate	0.9	high_dimension	0.9	feno	0.5	hospic	0.5
conjug	0.9	false_discovery_r	0.9	general_linear	0.9	tumor_shape	0.5	rasch	0.5
mcmc	0.9	synthet_data	0.9	asymptot	0.9	sbm_model	0.5	dif	0.5

Table 3: The list of 5 most distinctive stemmed words in the 10 unseeded topics. Stemmed words divided by "_" indicate n-grams.

Within this framework of seeded and unseeded topics, we applied seeded LDA to the pooled abstracts in each session. The traceplot raised no concerns about convergence. The total computing time was 70 seconds on the laptop previously described.

The calculation estimated the percentage of each session that is "drawn" from each of the available topics. In nearly all cases, a session has substantial percentages from a handful of topics, but small percentages from many topics. To regularize this outcome, we start with the topic having the smallest non-zero percentage, set that percentage to zero and reallocate its weight proportionally to the remaining topics. The reallocation process terminates when all remaining topics have percentages at least equal to 20%. This choice of threshold reflects a pragmatic judgement that if a topic participates in a session by less than 20%, then that topic is essentially incidental to the content of the session.

As usual with LDA, the analysis depends upon a large number of parameters, and it would be problematic if the results were sensitively dependent upon those choices. To examine this possibility, we performed a sensitivity analysis that varied the minimum number of times a token had to appear in order to be retained (3, 4, 5, 6; our analysis used 5), the length of n-gram (3, 4, 5, 6; our analysis used 5), and the significance probability cutoff needed for declaring an n-gram (0.05, 0.01, 0.005, 0.001; our analysis used 0.005).

Our comparisons are not based upon the perplexity (Blei et al., 2003), since that measure depends upon the number of tokens, and this sensitivity analysis changes that number. Also, our focus is upon interpretability of the topics in terms of the statistical discipline. So, instead, we compared the 20 most distinctive tokens in the each of the 51 topics across the 64 experimental conditions in our experiment.

For each of the 63 conditions that did not use the exact settings chosen in our analysis, we looked at the number of top-twenty distinctive tokens in common with our analysis among the topics found in these new variations. (Since the new topics are not labeled, they were matched, one-to-one, with original topics according to greatest agreement among their distinctive tokens.) Then we averaged the number

of common distinctive tokens over all 51 categories and all 63 runs. The average and the standard deviation were 10.21 and 0.380, indicating much agreement among the distinctive word lists. We feel this outcome shows that our original results are fairly insensitive to the parameter choices that we made. Supplementary material posted on-line contains much more detail about the sensitivity analysis.

4 Session Assignment Algorithm

The JSM schedule has 13 110-minute time bands when parallel sessions may occur. Normally, the sessions of the program are manually assigned to time bands by the program committee. Assignment is a labor-intensive and time-intensive process. The program chairs from all the ASA sections meet for two or three busy days, and that builds upon much prior work by the JSM program chair and three ASA staff members. At all steps of the process, everyone works together to try to minimize overlap in content. Nonetheless, anecdotally, many JSM attendees still complain about being forced to choose between two similar sessions in the same time band.

Clearly, an optimal solution is not unique. Interchanging the assignments for any two time bands having the same number of parallel sessions provides an equally good solution. And interchanging two sessions in different bands that both have the same amount of participation in the same topics provides an equivalent solution.

In exceptional cases (e.g., late-breaking sessions, memorial sessions), a person may give more than one talk at the JSM. And a person may also chair a session as well as speak. So an additional constraint on the scheduling is to ensure that no one is double-booked in the same time band. Finally, there is a soft constraint that sections with two or fewer guaranteed invited sessions not be scheduled on Sunday or Thursday, but this is often waived (based upon speaker availability, organizer request, or to avoid double-booking) and was not used in our scheduling.

Our assignment procedure has three phases:

1. Assign people with more than one role to sessions in different time bands.

2. Randomly assign the remaining sessions.

3. Greedily optimize the assignment.

Steps 2 and 3 are repeated 100 times, and then we use the best local optimum found. More repetitions might find slightly better optima, but a heuristic argument in Section 5 suggests the improvement would be negligible.

Let $\sigma = \{s_1, \ldots, s_N\}$ be the set of sessions that must be assigned to a band. And let $\mathbf{\Gamma}$ be the $N \times K$ matrix with entry $= \gamma_{ij}$ being the extent to which session i participates in topic j. As previously described, the regularization that zeroes out small topic weights ensures that many γ_{ij} equal zero and all non-zero entries are at least 0.2.

For two sessions s_i and s_j, their total variation distance is

$$\delta_{ij} = \frac{1}{2} \sum_{k=1}^{K} |\gamma_{ik} - \gamma_{jk}|,$$

so small values of δ_{ij} imply that the sessions have strongly overlapping content. To measure the topic overlap for an entire assignment of JSM sessions, we use

$$\rho = \sum_{i=1}^{N} \sum_{j=1}^{N} \delta_{ij} \theta_{ij}$$

where $\theta_{ij} = 1$ if s_i and s_j are assigned to the same time band, and otherwise it is zero. A good assignment produces a large value of ρ.

It is possible that the JSM Program Committee would like to modify this measure. For example, it might be that content overlap between an invited session and a contributed session is less problematic that overlap between two invited sessions. It would be easy to make that alteration, but absent any information on what the relative weights should be, this analysis puts all sessions on an equal footing.

Table 4 shows the 13 time bands that were originally planned for the JSM program, before the pandemic, and the number of parallel sessions in each. The number of parallel sessions in each time band is mandated, and is a constraint under which the Program Committee must work. The Sunday and Thursday bands have somewhat fewer sessions because the ASA has learned that fewer people attend on those days.

Time band	Day	From	To	# of parallel sessions
1	08-02-2020	14:00	15:50	38
2	08-02-2020	16:00	17:50	38
3	08-03-2020	08:30	10:20	40
4	08-03-2020	10:30	12:20	43
5	08-03-2020	14:00	15:50	43
6	08-04-2020	08:30	10:20	40
7	08-04-2020	10:30	12:20	43
8	08-04-2020	14:00	15:50	44
9	08-05-2020	08:30	10:20	41
10	08-05-2020	10:30	12:20	41
11	08-05-2020	14:00	15:50	40
12	08-06-2020	08:30	10:20	37
13	08-06-2020	10:30	12:20	39

Table 4: Schedule of JSM. The number of parallel sessions in each time band is fixed before the JSM Program Committee begins its work.

A literature search found no off-the-shelf optimization routine for this kind of scheduling problem, but it is clear that the problem will have many local maxima. So we use random restart hill-climbing, as recommended in Russell and Norvig (2002).

Our algorithm starts by assigning sessions with people who play multiple roles in the conference, to prevent double-booking. Then the remaining sessions are placed at random, respecting the number of parallel sessions required in each time band. That random assignment is subject to two additional constraints—no time band has two or more introductory overview lectures (IOLs), and IOLs should be scheduled on Monday, Tuesday or Wednesday. The algorithm then calculates ρ for that assignment.

Next, the algorithm randomly picks two time bands and swaps two sessions at random between them (without double-booking a participant or having more than one IOL). The algorithm calculates the new ρ. If it is larger than the previous one, the swap is kept; otherwise the swap reverts and a new swap is tried. The algorithm

terminates when 10,000 attempted swaps have not produced a larger ρ. Additional swaps might find some improvement, but our results indicate that this rough rule finds much-improved schedules. The total computing time was 290 seconds.

This algorithm might be improved by trying swaps such that the swap exchanges sessions that have small distances to other sessions in their time bands. This would accelerate the search, but at the cost of never (or rarely) moving sessions that are distant from other sessions in their time band, and thus not fully exploring the optimization landscape. There are obvious tricks that avoid this problem; e.g., usually choosing a session with small intraband distances, but sometimes trying a swap for a session that has large intraband distances. However, our simpler algorithm works well.

5 Optimizing the 2020 JSM Schedule

The best assignment obtained by the algorithm had ρ = 10,098.06. The original assignment made by the 2020 JSM organizing committee had ρ = 9,923.74. As a benchmark, the average ρ from 100 random allocations (but not allowing double-booked participants, nor two IOL sessions in the same time band) with no search for improvement was 9,891.31.

We do not know the minimum possible value of ρ among all session assignments respecting the constraints, nor do we know the true maximum, though it is certainly less than the 10,488 obtained from having no overlapping content in any time band. But to estimate the improvement due to the hill-climbing searches, we treat the average of the scores from the random assignments as the minimum and 10,488 as the maximum, and calculate the amount of improvement by $(\rho - \min)/(\max - \min)100\%$. Since the true maximum is surely less than 10,488, this underestimates the improvement percentage.

For the schedule originally planned, the improvement over averaged random assignment is 5.8%. For the optimized schedule based on minimizing topic overlap, the improvement over averaged random assignment is 37.1%. This is at least a six-fold

improvement over the intended schedule.

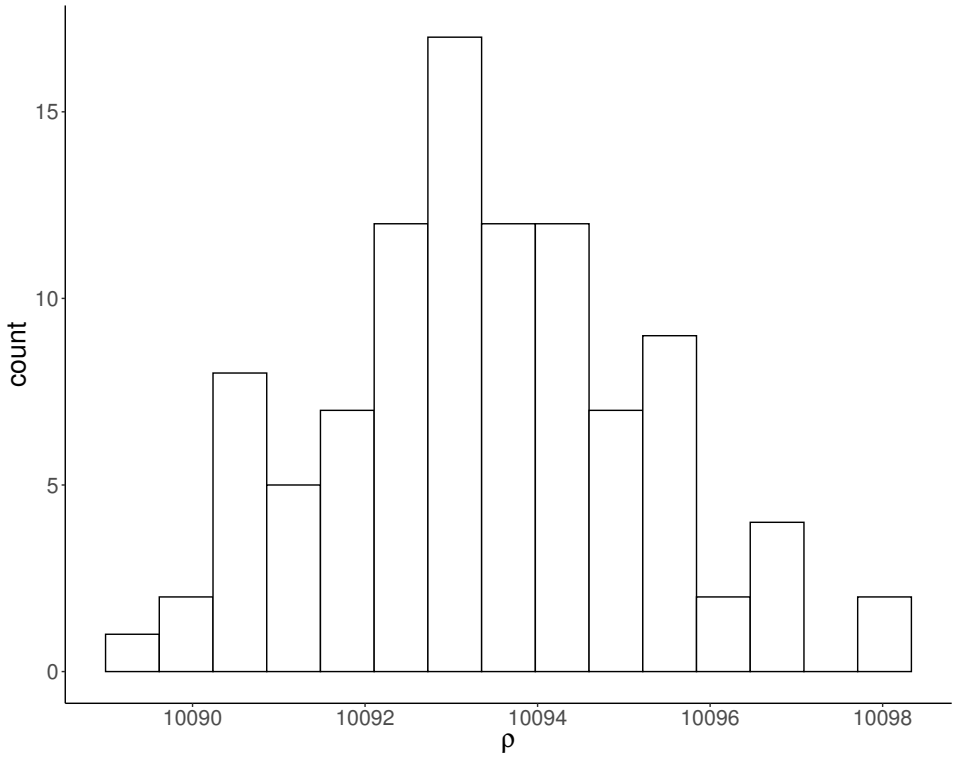

Figure 1: ρ histogram plot

Figure 1 shows the histogram of the local maxima found in our 100 random restart hill-climbing searches. It is approximately normal with mean 10093.38 and standard deviation 1.825. Our largest local maximum is 2.56 standard deviations above the mean, which suggests that further search would not find a meaningfully better maximum.

There is a body of work in probability theory which finds general conditions under which a function of permutations of real numbers is asymptotically normal (Porter, 2001; Hoeffding et al., 1951; Chatterjee and Diaconis, 2017). These results do not directly apply to our situation since (1) our swaps have restrictions, and (2) we use hill-climbing to find the local optima. The swap restrictions are probably negligible. Hill-climbing results seem much harder to prove, but we believe that for a very pocky surface such as this one, asymptotically normal behavior is plausible.

Table 5 compares the ρ values from our best schedule to that planned for the 2020

JSM. The mean for the average column is larger for the best schedule than that for 2020 JSM schedule, but it is worth noting that the assignment averages are more than three times less dispersed than the 2020 JSM averages (mean $\pm$ sd is 0.966 $\pm$ 0.004 for the best assignment and 0.950 $\pm$ 0.014 for the 2020 JSM assignment).

Time band	# of parallel sessions	Best assignment		JSM 2020	
		Sum	Average	Sum	Average
1	38	677.46	0.964	649.19	0.923
2	38	677.79	0.964	650.13	0.925
3	40	754.69	0.968	748.33	0.959
4	43	875.87	0.970	869.37	0.963
5	43	876.90	0.971	841.00	0.931
6	40	750.13	0.962	746.71	0.957
7	43	877.27	0.972	863.91	0.957
8	44	917.35	0.970	894.50	0.946
9	41	793.49	0.968	789.14	0.962
10	41	793.17	0.967	776.94	0.947
11	40	751.87	0.964	742.40	0.952
12	37	638.11	0.958	639.27	0.960
13	39	713.96	0.964	712.85	0.962
		10098.06		9923.74	

Table 5: Comparison of ρ values between our best assignment and the originally planned JSM 2020 schedule, broken out by time bands. Note that our algorithm's worst case is time band 12 (8:30 to 10:30 a.m. on Thursday).

A reviewer asked for more information on our worst time band, and we provide that below. But we also note that both the original program and the best solution found in our analysis are posted on-line as supplementary material for this paper.

Time band 12 is our worst. There were three sessions whose dominant topic was topic 13, Data Science Education. These were "Looking for a Nail to Hammer? Come Find Datasets from the Largest Clinical Trials Groups", "Teaching Data Sci-

ence for Good: How University-Based Initiatives are Shaping Future Statisticians' Societal Impact", and "Transforming Your Stumbling Blocks into Stepping Stones". Additionally, there three sessions for which topic 49 was dominant. Topic 49 was unseeded, but given its distinctive words, it is clearly related to the asymptotic behavior of high-dimensional linear models. Those sessions were "Sufficient Dimension Reduction and Variable Selection for High-Dimensional Inference", "Semiparametric Inference with High-Dimensional and Complex Data", and "High-Dimensional Statistical Inference Meets Large-Scale Optimization". The JSM Program Committee could use this information about the problematic time band to consider possible human-devised adjustments to reduce those overlaps.

6 Discussion

Every year, the ASA struggles to create a program for the JSM that minimizes content overlap in the same time band. Nonetheless, too much content overlap is persistently the chief complaint among attendees in the post-JSM satisfaction survey.

This paper describes a general methodology for minimizing overlapping content in complex professional society meetings with parallel sessions. It applies that methodology to the program that was originally planned for the 2020 Joint Statistical Meetings, but which was subsequently obviated by the pandemic when the meeting became virtual.

The key ideas in this work were to use seeded topic models to identify overlapping content in the same time band, and then to use an optimization algorithm to reschedule sessions so as to reduce that overlap. LDA always requires some tailoring to the application, but this analysis has made customary choices for the various parameters and in the pre-processing steps.

In our application to the Joint Statistical Meetings program, we obtained a much improved schedule. Compared to random assignment of sessions to time bands, the original program showed about a 6% improvement, whereas our method achieved at least a 37% improvement.

Given these results, we believe that the ASA could improve the JSM experience for its members by adopting the method we have described to schedule, at least as a preliminary draft, future JSMs. Additionally, use of our approach would significantly reduce the labor required of the members of the JSM Program Committee and save the ASA a significant amount of money by obviating the need for an in-person meeting to reconcile overlapping content in the program. No doubt future JSM organizers will want to perturb the algorithm's recommended schedule, but the quantitative measure of overlap will help them to assess the impact of their changes.

References

Kenneth Benoit, David Muhr, and Kohei Watanabe. *stopwords: Multilingual Stopword Lists*, 2019. URL `https://CRAN.R-project.org/package=stopwords`. R package version 1.0.

David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

Sourav Chatterjee and Persi Diaconis. A central limit theorem for a new statistic on permutations. *Indian Journal of Pure and Applied Mathematics*, 48(4):561–573, 2017.

Teague R Henry, David Banks, Derek Owens-Oas, and Christine Chai. Modeling community structure and topics in dynamic text networks. *Journal of Classification*, 36(2):322–349, 2019.

Wassily Hoeffding et al. A combinatorial central limit theorem. *Annals of Mathematical Statistics*, 22(4):558–566, 1951.

Jagadeesh Jagarlamudi, Hal Daumé III, and Raghavendra Udupa. Incorporating lexical priors into topic models. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 204–213, 2012.

Younghoon Kim and Kyuseok Shim. Twilite: A recommendation system for twitter using a probabilistic model based on latent dirichlet allocation. *Information Systems*, 42:59–77, 2014.

Martin F Porter. Snowball: A language for stemming algorithms, 2001.

Stuart Russell and Peter Norvig. Artificial intelligence: a modern approach. 2002.

Grigori Sidorov. *Syntactic n-grams in computational linguistics*. Springer, New York, NY, 2019.

Jacopo Soriano, Timothy Au, and David Banks. Text mining in computational advertising. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 6(4):273–285, 2013.