Tips for the PhD in Empirical Financial Economics:

From Developing a Research Idea to Getting It Published

Jens Jackwerth

March 26, 2022

Abstract

This practical guide is for the PhD student in empirical financial economics. Tips range from finding and developing a research idea to collecting data to choosing programming languages and methods to writing a paper to presenting and publishing it.

Keywords: Financial economics, data collection, writing advice, research tips, PhD students

JEL: G10

Jens Jackwerth is from the University of Konstanz, PO Box 134, 78457 Konstanz, Germany, Tel.: +49-(0)7531-88-2196, Email: jens.jackwerth@unikonstanz.de. I thank Ben Golez, Joachim Grammig, Jim Hodder, Natascha Jankowski, Manuel Leininger, and Alberto Plazzi for helpful comments. Cindy Gross (http://cmgediting.com/) provided excellent copy editing and a wealth of suggestions and improvements.

Tips for the PhD in Empirical Financial Economics:

From Developing a Research Idea to Getting It Published

Contents

Introduction

Many PhD students are initially at a loss about what their next few years are all about. A PhD is about communicating self-guided research. Let me explain those last three terms. Research is to pose questions about financial economics and to answer them. Those questions need to be interesting to you and relevant to the profession. Answers come from the analysis of data. Research is a painstaking and long process, so you better enjoy it.

You are your own guide in designing your research and later communicating it. No professor will tell you what to do exactly, and, if you so desire, you can shirk on your work. Thus, beware of demands on your time (social media, requests to work for others). Set yourself clear, small goals for each day. Then sit down and do them. Structure your days (say, mornings for creative work, afternoons for tutorials and free time). Stick to a time line while allowing for slippage; research always takes longer than anticipated. Likely the most important ingredient to a PhD, before raw intelligence, is dogged persistence.

Finally, communicate your self-guided research through writing papers, presenting your work, and teaching. You only get credit after successful communication, and writing good papers is the key to success. A good paper engages the reader, convinces them of the relevance of your research question, and delights them with your results. Necessary, careful writing is difficult and takes much work. Lifelong work on your writing is as much part of your academic career as lifelong work on your research.

A small literature provides helpful insight into the first steps toward a PhD. I will concentrate on my own field of empirical financial economics and borrow liberally from the aforementioned literature. For more tips on writing in finance, consult Cochrane (2005); for general writing, consult the references below in the section on Writing. Gentzkow and Shapiro (2014) focus on data and programs, while Skrainka (2010) deals exclusively with programs. This text is broader, covering some thoughts about generating research ideas, how to turn those ideas into papers, and how to get those papers published.

Research Question

Few PhD students start their PhD with a clear notion of what their first paper will be about. But many are curious. This curiosity can be used to the student's advantage. Channel that curiosity into developing research questions. Pick an area about which you are curious (hedge funds, say) and where you or your university has sufficient data. Your curiosity should last through the three papers that you will write in this area. Otherwise, find another area. Within any area, many open questions are waiting to be answered. You can later choose questions within the area according to your particular interests (empirical, theoretical, statistical, behavioral).

Stay in one area for the PhD. Entering each new area requires about one year before you know what you are doing. If you write papers in two areas, you need two years to get ready. That is often too long.

Generating Research Questions

For the first paper, get your feet wet quickly rather than agonize forever on the perfect research question. Save those agonies for the job market paper. A master thesis can possibly be turned into a paper. Or an adviser might suggest whether a particular area offers some halfway interesting research question. Or you can develop a class project from your PhD course work into a paper. Otherwise, read recent survey articles as they often contain ideas about open questions. The *Annual Review of Financial Economics* is a great source of such articles.

Once you have chosen a topic, read, read, read. Read the abstracts in the *Journal of Finance*, the *Journal of Financial Economics*, the *Review of Financial Studies*, then further down the list as far as you can take it, and subscribe to the relevant mailings at SSRN. Each topic has its own lingo and rules. Learn them to easily navigate the topic. For example, in hedge funds, the main factor model is Fung and Hsieh (2004). Every paper implements standard filters for hedge fund returns: use dead hedge funds (ones that no longer exist in the database) but skip the first 12 returns. And so on. Unless a senior colleague (adviser or coauthor) explains those rules to you, you need to find them by reading recent research papers. Once you know the rules, pass them along to others. Pay attention to common choices and take note of them: OLS is acceptable, but researchers combine several hedge fund databases nowadays. Some papers skip more than the first 12 returns, some skip the last 12 returns, too. And so on.

For the first paper, stay close to existing work. Add a new factor and see whether it explains hedge fund returns better. Condition alpha (the skill of the hedge fund manager) on some interesting variable. Does that improve your factor model? These examples are easily doable, will teach you how to wrangle the data, and will prepare you for more challenging work later. Note that only the job market paper needs to stand out (and if you go into industry, not even that). So a less ambitious first paper can still go into your thesis if it is solid work.

Once you are a little more comfortable with a particular research area, develop your own research questions. Take notes whenever you come across something interesting (send yourself an email, a social media message, or write on an index card). Organize those notes in some ideas folder (say, \ideas\hedge_fund_factor_models) and put literature into another subfolder (\ideas\hedge_fund_factor_models\literature). Sketch out these ideas as you file them in form of a (digital) journal. The ideas folder will be a great resource for research questions and, in case of less challenging and interesting questions, for topics to hand out as master theses or class projects.

Research exists along a continuum. On one end, stay close to existing research and adhere to all the conventions and rules of the area. Here, methods are well established, and work is incremental. That is the reference point for your first paper (add a factor to a standard factor model). More interesting and challenging questions require you to break some conventions and rules. Possibly new methods are needed (factor models might not be of any use in hedge fund research because of statistical limitations). The most outrageous questions break completely from the existing paradigm and mark the other end of the continuum.

For your second and third papers, try to leave the comfort of staying close to existing research. Try to think more broadly about what is really interesting or really puzzling in your area. These more ambitious questions follow from brainstorming, where you leave behind what everybody else has been doing. Do not think about the next factor to add to a factor model but about replacing factor models altogether. You might not find a good answer (nobody else has so far) but asking good questions opens the horizon. Keep on asking questions and try to arrive at one where your answer is novel and interesting.

Developing Research Questions

Develop a few research questions, and then weed them out. This process normally involves an economic argument about what one should find in the data according to economic theory and then checking whether that effect actually can be found in the data.

Once you come up with an interesting economic mechanism, run it by trusted colleagues, other PhDs, and your adviser. If it generates head scratching, think again. Listen to critiques and try to convince yourself of the viability of the project. Next (or ideally already before you communicate your idea), work out how to econometrically show your anticipated result. Use the simplest econometric technique that gets the job done. Readers and referees have little patience for complicated methods they do not understand, unless they are convinced that those methods are needed. Start with OLS unless you really need a nonparametric kernel regression. Interpolate linearly before you explain a B-spline.

What does your method (say, OLS) require of the data, and are those requirements met? Can you assume normally distributed errors? For hedge fund returns, errors are often not perfectly normally distributed, but it is still accepted to use OLS. Is the error uncorrelated with the explanatory variables? How do you identify your variables of interest? Which data features allow you to measure the desired effect? For hedge funds, a great idea might involve holdings data, but, except for very coarse 13F data, one typically cannot obtain hedge fund holdings. What econometric problems are expected (reverse causality, non-normal errors), and how can you solve them? Address these questions before spending too much time collecting data.

Ideally, you have some small, easily accessible data set to check your main idea and determine whether the data generate the desired effect. If not, you can still write a negative result and use that as one of your three papers in the thesis, but publishing a negative result is difficult to do. Editors much prefer new findings as opposed to negative results. That is lamentable, but it is what you are facing. Thus, angle for a question where you can document an effect.

Note the stability of your results. If small variations in the methodology (different samples, using OLS or GLS, filtering data one way or the other) change results, then be wary. You might be able to rig up a paper where results are just the way they are supposed to be, but your house of cards collapses when the first referee asks for a robustness check you did not anticipate. Rather, try to find more stable effects that hold in many different robustness checks.

Coauthors

Starting as a research assistant is a great way to learn the ropes of a new research area, but it does not automatically generate coauthorship. Ideally, become the junior partner on a joint paper with a more experienced colleague or your adviser. However, doing so runs the danger that readers will assume that the adviser supplied the idea, and you did all the programming. Joint papers with another PhD student or junior colleague suffer less from this taint but forgo the advantage of having a senior adviser. A third option, the sole-authored paper, completely does away with the stigma and could be the best option for a job market paper.

In general, two is a good number of coauthors. Two coauthors can share the work, can internalize feedback and critiques, and allow for specialization (programming, writing, logical structure, generating research questions). Most colleagues will value a coauthored paper almost as highly as a single-authored one, as long as you have at least one single-authored paper to your name to establish the fact that you can work on your own.

Find coauthors by commenting on papers and presentations. Send a carefully thought-out comment or question on a paper to the author and develop a relationship with them. Then air an idea — you will quickly sense if the colleague is interested in becoming a coauthor. Be careful in your choice of coauthors; you could be spending more time with them than with your spouse or partner. Ending a collaboration can be difficult to do, as you have your joint paper to take care of for years to come.

Coauthor relationships are like any other meaningful relationship in your life. They are long-lasting and can be incredibly rewarding, yet require much work and care over the years. Timing matters, too. A coauthor might be a great fit for one topic, but not the other. Both you and the coauthor change over time; the relationship needs to be adjusted accordingly.

Papers with more than two coauthors often turn unwieldy and are prone to shirking. They often require one colleague to manage them (distribute work, integrate results, set an agenda, and keep the project up and running). If you do not know who that person is, then it is likely that the person does not exist. This does not bode well for the project. Given your tight time line of three or four years to job market, avoid unwieldy projects.

Data

Starting with standard data speeds up your progress. So use an existing hedge fund database for the first paper rather than hand-collect your own data. Once you know your field better, think about collecting additional data on some interesting aspect (few researchers hand-collect LinkedIn data of hedge fund managers in conjunction with hedge fund databases, but maybe managerial characteristics explain some aspect of hedge fund returns?).

Keep your eyes open for unusual data. Researchers have used rarely used SEC-required filings on hedge funds; they have conducted interviews with hedge fund managers; they have talked companies out of proprietary due diligence data on hedge funds; and they have looked at government registers of the employment history of hedge fund managers. If you find something on a website, can you legally scrape it and use it for your purposes?

Collection

How much data to collect? If in doubt, take what you can get. More likely than not, you will want to control later for additional variables or want to investigate some new effect. Going back and getting more data is a hassle. Get it now.

Data can change over time or might simply no longer be available in the future. Get it now.

Storage and Data Security

Store your data in the raw format as you downloaded it. Carefully document the data, where it is from, and when it was downloaded. Then save it in several places. A copy on a portable hard drive under your desk will not suffice if your whole house burns down. Researchers at UC Berkeley learned that lesson the hard way when Oakland Hills burned down in 1991. Ideally, use several cloud services with good automatic backup. Use offline systems at remote places for periodic updates in case the main system gets hacked and scrambled.

Thereafter, do not ever touch those files again. This allows you to go back to the original data for replication, backup, or restarting your work if you made a hash of things. Make copies of raw data and work on those copies.

The following data sections borrow freely from Gentzkow and Shapiro (2014) and Skrainka (2010).

Computer Language

I use open-source Python; many colleagues use proprietary Matlab. Coding in such widely used languages makes collaboration much easier. The number of ready-made modules (regressions, machine learning) seems larger in Python and is quickly expanding. If you start coding, I would go for Python, which also allows using econometric models written in R. Be warned that Python is not completely backward compatible. In rare instances, older code might not run on the latest Python version. If you already know Matlab, fine. Both languages are interpreted, so the computer reads your program anew every time you run it and translates it "on-the-fly" into machine-readable instructions. This works well for finance research, where you change the program frequently and execute the final version once for the published paper.

In compiled languages like C++, programs are first translated once into machine-readable instructions. These machine-readable instructions then execute much faster on each run. This is more important for industrial applications that need to run unchanged many times (say, the back-office functions of a hedge fund). But the compilation takes time and is required each time you change the program. As compiled languages tend to be more difficult than interpreted languages (e.g., data types need to be specified for each variable), I do not recommend them. If you already started with C++, fine.

One small exception: if your program has some very expensive computation at the core of it that is executed a zillion times (say, a complicated numerical fit inside a huge simulation), then you can program only that inner problem in fast and compiled C++ and call it from your interpreted Python program.

Learn your programming language properly. Like with any foreign language, just being able to say “Hello, one coffee, please!” will only get you exactly that. If you want to get more out of your programming, you need to know more. Yes, understanding classes, tuples, and DataFrames in Python is part of that.

Programming

Write clear and understandable programs. Use a space to separate instructions (“xr = r – rf”, not “xr=r–rf”). Comment as much as you possibly can: at the beginning of each program, state the date, author, what the program does, which inputs are needed, and which outputs are generated. A change history goes here, too. Mention what resources the program needs (computing environment, files needed). Use short lines of instructions, and include only one step of the program per line. And if each line has a comment on what it does, great. It will help you when you go back and rework the program after some pause. In particular, comment anything difficult or any nonstandard behavior (say, a standard Python vector has elements 0 to $n - 1$ by which you index. If you use a vector of length n instead and index from 1 to n, you better comment that).

Any part of the program being used more than once should be written as a function. This is safer than copying the instructions and adapting it to the new requirements. One often forgets to change copied instructions in all the right places. Also, any part that is longer (starting at, say, 30 lines) should be turned into a function. It makes your main code shorter and simpler. Mind you: calling functions is slow, so do not go overboard. By extension, any number used more than once should be defined as a constant. Any table column (DataFrame column in Python) should have an obvious name. Using hardcoded column numbers to pick a column can stop working when you insert or delete columns. A named column keeps its name.

As an example, let us investigate factor models for hedge fund returns. Thus, write a function that takes as input the return data and a specification of what the right-hand side factors are. The function then regresses returns on factors. Use a descriptive verb to name the function: regress() would do. This function can then be variously called (say, regress(HF_data, [market]) for a market model. The function returns the regression results as output. Changes and improvements to regress() are thereafter correctly used whenever you call the function in your program. Do not use global variables but always pass information to the function (data, factors) and take back the output from the function.

If many functions are cluttering your main program, organize them by topic into libraries. Those can be shared across projects (many different projects all need a regression module). Rely on existing modules that are available with the programming language; there is little need to reprogram OLS. But it makes sense to have a special function to run OLS regressions on data from your personal hedge fund return database.

Stick to a style sheet (also called a coding convention): Write down how you call things. Names should be descriptive. A function uses small letters as in regress(). A variable uses underscore as in df_HF_return. Note the use of "df" to indicate the type of variable used (here a DataFrame). Label all variables according to type and you know which functions and methods work on each variable. There is a comment at the end of each line or before a group of connected lines. There is a space between any set of connected lines (think of a sentence). There is a line (---------) between blocks of instructions (think of a paragraph). There is a double line (==========) between sets of blocks (think of a chapter or section). Use "#%%" in Python ("%%" in Matlab) to create cells within your program that you can execute separately. Then you do not need to execute the whole program for each little change in the cell. Keep writing your own style sheet.

To keep your program running, try not to use outside programs or functions that can change (a colleague changing a function you use, a company changing the API that you use). Write unit tests that try out functions: test the function regress() by running it on collinear right-hand side factors. Run regress() on empty data or no data. Make sure that the resulting behavior is what you had in mind (program stop? Error message? Dump this regression but continue the rest of the program?). Use the debugger. After debugging, output should go to file and no longer to screen (both logging information and results).

Before optimizing a program for speed, get it to run correctly. Then optimize your program by timing it first with a profiler. See where your program spends most time. Can you speed up those parts?

Create a fast version of your program that ideally computes in a few seconds (say, only use every 10th observation). If the fast setting gives you results comparable to the full run, then debug and develop in the fast setting. Finally, run the full version overnight to make sure you still have your result. This greatly speeds up the development process. The wait is much shorter when rerunning the program after some small change to it.

Always use the built-in functionality of Python before you code something yourself. Built-in programs are faster, cleaner, and less buggy. Programs in Python libraries also tend to be carefully vetted before inclusion. Beware of other programs on the web; your results should not depend on some stranger's dubious code. And even a professor of statistics posting a program on the web is a stranger unless you really know that person. What is worse, even good programs might still not work for you if they have strange defaults (different areas of science use different conventions and defaults) or make implicit assumptions buried deep in the program. This problem extends to the high-quality libraries of your programming language. Even getting exactly the same ridge regression output from Python and Matlab can be challenging as some scaling conventions vary slightly. Read the freaking manual!

Thus, use programs on the web as inspiration and write them yourself. Nothing teaches you the workings of some program better than rewriting it yourself. This extends to industrial strength software: write your own regression program to appreciate how it works. Then compare your results to *sklearn.linear_model.LinearRegression()* and make sure that they are the same for simple cases. In the end, rely on the library for the heavy lifting. Beware of black boxes that you do not understand (say, machine learning programs). How can you stand by results that you cannot explain nor interpret nor defend?

Avoid loops as they are slow. Often, you can rewrite the program to operate on a whole vector of numbers instead of looping through the vector and operating on every number in turn. Take logs of a matrix with all returns for all hedge funds instead of looping through all hedge funds and all dates to take the log of each return. This works best for mathematical functions. Look at every single loop and see if you can vectorize it.

Relational Database

Organize your raw data without redundant information, same as you would write programs without redundancy. Here is an example of hedge fund data:

Date	Return	HF_name	HF_style
202111	0.03	Jackwerth	equity
202112	-0.05	Jackwerth	equity
202111	0.10	Miller	fixed_income
202112	0.02	Miller	fixed_income

In such a data set, fund information appears multiple times and is thus redundant. Changing it everywhere requires careful work in multiple cells (say, because of an update or a correction of a mistake). It is much better to split the data using a unique numerical identifier (HF_ID) into two tables. Then each piece of information appears only once. Once you change it, you can be sure that you did not forget an update somewhere else in the database. In database speak, the identifier is called a "key," and the splitting is called "normalizing the database". The split results in a return table and a table for hedge fund information:

Date	Return	HF_ID
202111	0.03	420
202112	-0.05	420
202111	0.10	666
202112	0.02	666

HF_ID	HF_name	HF_style
420	Jackwerth	equity
666	Miller	fixed_income

This collection of nonredundant tables is called a "relational database." Any change to the data now occurs in only one place. We can have many such tables: add to the hedge fund information table a further key "employee_ID" and another table with employee information (say, employee_ID, name, salary).

Filtering

Raw data need attention. Once tables are normalized, correct mistakes, eliminate parts not needed, and change the data as needed (say, divide by 100 to convert percentages to returns). Statistical software normally does not run on normalized tables. Thus, for analysis, merge (join in database speak) the data as needed, but all changed information is then correctly duplicated.

All this work should be organized in one computer program; I always call mine "batch.py". I thus know that once I run batch.py, I have all data processed in order, and I have a properly documented account of how I did that. If I need to replicate my data work, I run batch.py. If batch.py gets too large, then chop the tasks into more manageable programs and have batch.py call a set of subroutines (e.g., clean_data.py, merge_data.py).

Put batch.py into a project folder with subfolders \input, an empty \output, \subroutines, and an empty \temp. A file readme.txt (in the project folder) will inform you in the future that the raw data goes into \input, that batch.py calls the subroutines (in \subroutines) in their correct order, and that output.csv is put into \output at the end. With such clean subfolder structure, write batch.py such that it calls files relative to the local directory (../input/raw.csv instead of C:/Jens_research/paper17/input/raw.csv). Then the whole folder can be moved around but batch.py will always work. Setting up the folder in this way ensures that you can replicate at later points the filtered data from the raw data. Also, having all these steps automated saves you much work if the raw data get updated and you need to rerun all filters. See the painful retraction of a *Journal of Finance* article as a case study of what can happen if you cannot replicate your own work, Rampini, Viswanathan, and Vuillemey (2021).

Here are some variations on this batch idea. If some processing step is superbly expensive in terms of computing power or manual labor, one can consider splitting the procedure: have one project folder with batch1.py to go from raw data to level_1 data (say this is the expensive bit), and another project folder with batch2.py to convert level_1 to level_2 data (say, this goes much faster). Run batch1.py (and then batch2.py to finish) only if really needed (say, there is new raw data) and run batch2.py as often as desired as long as the level_1 data does not change.

Versioning

Over time, programs need updating. In the above setting, changing programs is fairly straightforward. Thereafter, always run batch.py to make sure that the correct output.csv has been generated.

Changing code means that the new code might no longer work. Thus, working alone, it might suffice to count versions and call the files V01batch.py, V01clean_data.py, V01merge_data.py, and V01output.csv. Once programs change, you call them V02batch.py and so on. You keep the old V01batch.py as backup.

This system even works for two coauthors in a shared folder (say, on Dropbox) as long as they hand the "master" back and forth. So there needs to be communication that "today I work on the files in the project folder", I release the "master," and "my coauthor Ben can work on the files." Collisions are limited, and many papers were written this way. Yet, it bears the risk of colleagues changing different files simultaneously, so that the files no longer work together, or, even worse, both changing the same file (Dropbox will flag such conflicts) and having to sort out which copy to keep and which to delete. Work with more colleagues and collisions multiply.

Versioning systems, such as Git (or, say, SubVersion), are much safer. These programs allow you to have your folders in the cloud (the actual website is called GitHub, allows you to set up a free account, and uses Git), check them out for local work, run the batch.py to make sure the new version does what it is supposed to do, and check back the changed folder. The software keeps track of all versions for all colleagues working on the same project. With conflicts (a colleague changed the main program while you changed it, too, but based on an older version of the main program), Git will warn you, force you to update to the current main program, check that your changes still work, and then allow you to integrate your changes to the main program. Branching allows for experimental programs; merging allows for integrating successful new ideas into the main code. Going back to the last stable version is as easy as using the undo-button. The benefits outweigh the costs in terms of learning to use the versioning software.

Git allows for some task management, where colleagues are assigned tasks, can sign off on them, and other colleagues are informed about those actions. More advanced task management systems (such as Jira) are normally not required to write finance papers but are the reserve of large corporate software projects.

Structure of the Analysis

With the data in normalized tables, the analysis can begin. The programming of your results and the writing of your paper are two sides of the same coin. Work on both at the same time, and go back and forth between the two. I split the discussion into Structure of the Analysis versus Paper only to organize this text. Much guidance on how to choose and organize your results will follow from the structure of the paper, so read those parts as well before programming.

The moment you move beyond the explorative stage of a new paper, start writing the paper. Initially, this will be just a record of ideas, literature references, initial results, to do’s, and documentation of data. Use bullet points to be fast and to minimize the time lost if you abandon the project. As you progress, gradually shape this into a paper, following the outline of a paper and paying more attention to writing and completeness.

Again, like the batch.py file for the data generation, there should be a batch program for the results (project_batch.py). Inside, each and every result of the paper (not only figures and tables but also individual numbers needed in the text, such as a statistic or a worked example) should be computed in turn. Begin the program with a series of switches to toggle: F1 = 1, T2 = 0. Then generate Figure 1 only if(F1 == 0): and so on. Switching off Table 2 will keep the instructions but not generate the result. If you write all results to files with obvious names (F1.pdf, T1.csv), then you can point your Tex file to those files and you have a completely updated paper on the push of a button. The statistical R package has similar capabilities of writing your paper text in R and interlacing it with the results from your R programs.

Organize each result as a cell of the program (separate it by "#%%" in Python) so that you can run it separately from the rest of the program. Then you can update one table or figure at a time. Thus, in the end, you can generate the current version of your paper by running batch.py to process the data and project_batch.py to generate the results. Two files only and you are done.

Use functions to eliminate redundant instructions. Two nice functions to write are make_figure(data, parameters) and make_table(data, parameters), which do just that by reading the data (say from some regression output, reshape it into a formatted table, and write it to T1.csv).

Check your results against common sense and economic insight. An alpha of 400% might hide a factor of 100 – did you use basis points instead? In our factor models of hedge fund returns, alpha should be very noisy, the market beta should be noisy and around 0.3 on average, and all other betas should be noisy and often insignificant. Replicate results from the literature to check your code. Time permitting, have your coauthor code the main result independently, and compare notes.

Paper

Once you have the results ready, turn to structuring your paper. At this stage, I comment only sparingly on writing, which I cover in the following section at length.

Stick to the usual outline of a research paper. Readers and referees are busy and skim. Unless they are very interested, they will read the abstract, maybe the first pages of the introduction, then glance at the conclusion to see whether they missed anything. Cater to that and focus on those pages. Any creative way of setting up a paper confuses everyone.

Start with a generic outline (Title, Abstract, Introduction, Literature, Methodology, Data, Results, Robustness, Conclusion, Appendices) and fill it with bullet points. Label to do's clearly and with names of coauthors (Data, [TBC: Ben about the hedge fund database] [TBC: Jens on new factor data]). I find it helpful to have items color-coded for each coauthor.

Never copy and paste the exact same wording into a different section (say, main findings from Results to Introduction and to Conclusion). Annoyed editors and referees may reject your paper, calling it "too repetitive." Knowing how to say the same thing different ways is a useful skill you will need to build over your career. Unique wording makes your manuscript stand out.

Abstract

Try to distill your work into a proto-abstract. What is your key insight? Explain it in a few sentences, and keep the 100-word limit for an abstract that most journals enforce.

Pay attention to writing: use present tense and active voice ("I collect" instead of "data was collected"); short sentences; few adjectives, adverbs, and superlatives (avoid "novel," "very," "innovative"); and no "5-dollar" words (use "difficult" instead of "obstreperous"). At times you need to deviate but do so sparingly and convince yourself that you really ought to do so to make your point.

Journals often require that you provide JEL codes and keywords. My impression is that nobody uses either, so just get on with it. JEL codes stand for *Journal of Economic Literature* and you can look up the list here: https://www.aeaweb.org/econlit/jelCodes.php?view=jel

We are in category G, Financial Economics. Pick three JEL codes that fit and avoid meaningless ones (the ones that end in “0” and the ones with only one number after the letter). If your work touched on adjacent areas (econometrics, microeconomics), then you can use one of their codes. For the keywords simply scan the title and the abstract for five short phrases that describe the paper (hedge funds, factor models, factor zoo, regression, out-of-sample performance).

Title

Like your abstract, your title should leave an impression on readers, while communicating key points about your paper. The title is a yet shorter version of the abstract. Zero in on a concise, specific, and informative title. Ideally, your title uses and repeats key words that occur in your paper (hedge fund, factor model). Readers often scan only the title to determine whether they want to read the abstract and further on.

“The Illiquidity Factor in Models of Hedge Fund Returns: Overcoming the Omitted Factor Problem” would do in our example. In case of subtitles, the main title must be able to standalone. Here, “The Illiquidity Factor in Models of Hedge Fund Returns” would be even better unless you want to stress your contribution to the problem of omitted factors.

“Thoughts on Asset Pricing” would be too broad and bland. “A Tale of Two Factor Models” shows that you know your Dickens but verges on being pithy. Avoid such titles.

Do not use non-ASCII characters in the title or in your name. Many computers cannot handle them, and their use creates confusion. Also, do not change your title or name without need. Readers and referees connect the two and are confused if you suddenly sport a new middle name or a different title: Is that still you? Is that the same paper as the old one?

Introduction

The introduction (three to five pages) should sail quickly to the main point of the paper. No general thoughts about the meaning of life here or about the literature leading up to your work. Get to the point in very few sentences. And yes, this is the hardest part of the whole paper. Work on it. It is the hook of your paper, and you need to capture the attention of your reader. Read out loud the introduction, record it, and listen to it. Then you will immediately see how much work is still needed to weed out redundancies and to get to the point.

For our hedge fund factor models, this might work: "Hedge fund factor models only explain about one-third of return variation. Adding a new factor explains two-thirds of the variation. It . . . ." The first sentence immediately captures the attention of a hedge fund researcher as it describes a well-known problem. And you can do much better. Interesting. Let me read more about what you did!

Then describe the main research question; explain why it is interesting; and possibly hint already at the result. Next, provide a short overview of the method (without details, in particular if the method is standard, say, panel regression). Highlight problems that any reasonable reader has at this early stage (say, if causality is an obvious problem, how do you deal with that?). By page three, the reader should have a clear idea about what you wanted to look at, should agree that this is an interesting question, see how you answered it, appreciate that you took care of obvious pitfalls, and know the answer to your main research question.

Try to not let the reader off the hook: have each paragraph build on the last. If you are really ambitious, try a miniature cliffhanger at the end of each paragraph to make the reader eager to dive into the next. At the end of, say, a paragraph on your raw data, this can be as little as "Some of the observations were obviously wrong." The reader will likely read the following paragraph to find out what this means and how you dealt with it.

The "boilerplate paragraph" at the end of the introduction ("The paper presents the methodology in Section 2, the data follow in . . .") is not needed unless the journal requires it. More people have it rather than not. Have it and delete if strapped for space.

Literature

Collect all literature in a subfolder \lit and save them there as files (e.g., Jackwerth_2000.pdf). When new versions are available, move the old one to \lit\old and store the new one in \lit. If you have copious notes on a paper, put those into Jackwerth_2000.docx. Start this work early and file away whatever research you come across that might be relevant for your paper. This process almost automatically generates a good base for your literature section.

Cite the relevant literature and position your own work in relation to that literature. Do not provide a survey of the literature that distracts from your work. Here, you can also showcase your contributions and relate them to previous work. In that case, integrate the literature at the end of the introduction – if you have a separate Literature section, you almost invite the reader to skip it altogether, in which case your contribution part is lost. There is no need to snipe at other papers; the authors could well be your future referees. Also, given the level of noise in refereeing, you do not need to strategically cover every paper under the sun. Two pages should suffice.

Cite important works and describe their findings/insights in your own words. Do not quote verbatim unless the quote is so singularly funny or important that it needs to be put into the text. In that case, use quotation marks and cite properly. This should only ever happen for quoting Mark Twain. Then again, most finance papers are better off without Mark Twain.

Methodology

At this point of the paper, you essentially start the paper again, but now with details. Still, get to the main result as fast as possible. Methodology describes how you got there and might include a model. If the model is obvious (factor model of returns), then keep it really short. KISS your paper: Keep It Simple, Stupid. Use the simplest model/methodology needed to get to your result. If you are having more than one distinct methodology, separate them by subsection (same goes for results in that case).

Beware of the TLC fallacy: Tender Loving Care for some complicated method you devised in much hard work. If it is no longer needed, then chuck it out and go with something simpler. Complicated extensions can be mentioned ("We also estimate our model using GLS instead of OLS and results do not change much, see Section Robustness") and dealt with after the main result.

Provide the economic arguments that generate your results. It will often involve some econometric model. Check that the required assumptions for the model/method are in place (say, errors are normal enough so that you can use OLS). Check on reverse causality and explain how you deal with that. How do you identify your variables of interest? Convince the readers that the methodology is sound. That generates trust in your results.

For structuring the paper, consider writing out each hypothesis as such:

Hypothesis 1: Adding the new factor improves the explanatory power (adjusted *R*-squared) of the base factor model of hedge fund returns

An economic argument leads up to this hypothesis, a description of some methodology shows how you test the hypothesis. You can repeat the hypotheses in the Results section to guide that discussion, too. Once you did all this and the paper is ready, think about removing this clunky, formal writing. But keep the overall structure when you rewrite the hypotheses as plain text.

The gold standard for describing the methodology (and by extension, data and results) is that a good PhD student should be able to replicate your work. Thus, unless you provide programs and raw data, you need to be clear and exhaustive in your description. Move technical details into the (internet) appendices to avoid cluttering the main text.

You may use subsections to structure the paper, but not subsubsections.

Data

Provide essential information about your data (what it is, where you got it from, hand-collected or freely available or purchased) plus a few essential descriptive statistics. Anything else (details on the data collection, details on filtering, details on reworking the data) can go to an (internet) appendix. Keep the data section short; it can often be done in a page (using hedge fund databases does not require much additional information – colleagues know the databases, know the filtering).

Straightforward data go best after the methodology. If, however, problems of the data (noise in the data, peculiar error structures) cause choices in the methodology (a model with measurement error, an autoregressive moving average [ARMA] model), then flip the section order. Present the messy data first, highlight the problems, and then introduce the methodology to address those problems.

Results

Now state your base case result. Ideally, this is one table or figure. Note that each table or figure needs a title, a self-explanatory header, needs to be readable in black and white print and on small screens, needs labels on all axes (figures) and in the first column/first row (tables), needs to be understandable on its own, and needs some discussion in the text. Spend time on interpreting this main result and guide the reader through the table/figure so that the reader can follow your argument arriving at your main result. What is the statistical significance (*p*-value, *t*-statistic)? What is the economic significance? Given enough data, you will often achieve the earlier, but you also need to address the latter.

Next, repeat the result for other important cases that are meaningfully different from the base case. This is about economics: Split the sample into recessions and expansions and discuss your results over the business cycle. If results differ from the main result, put that here and discuss in detail. Cutting your sample in the middle to show that your estimates do not change wildly has little economic insight and goes into Robustness.

Robustness

Less relevant cases go here, including all changes to the methodology. In particular, put cases into Robustness where the result does not change much. Cases in which the result hardly changes at all can be described as such with the necessary table or figure in the (internet) appendix.

Conclusion

As few readers ever get to the conclusion, keep it short. It is a combination of abstract and introduction, thus summarizing the paper yet again in one page. Do not just cut and paste: some referees skim the abstract and the intro and then go for the conclusion to see whether they missed something but skip (most of) the rest.

Appendices

You are now done with the paper. Check the page number. It should be lower than 30 if you have tables and figures in the main text and lower than 25 if you put them after the references. Readers love short papers but freeze when seeing a page count of 100. I personally like tables and figures in the text to have them close to where the text covers them. Journals typically require them at the end of the final version for typesetting.

If the paper is too long, shorten it by getting rid of fluff. Then move excess material to the (internet) appendices according to its importance. The most important excess material goes into the appendix and, together with the main text, constitutes the printed version (as a pdf or journal article). Beyond that, there are separate internet appendices. Push less important material into internet appendices, and, finally, just delete the ancient rubble that made it through 50 iterations of the paper but is no longer needed.

Programs are not part of the paper. It slowly becomes more common to provide the programs that generated your results. This greatly helps interested readers in replicating the work. Put the programs onto your website once your paper has been accepted at a journal. You do not want somebody to take your programs, rerun your results, and submit it to another journal. Often you are not allowed or do not want to publish your raw data. Then do not. Rather, provide processed data after publication.

For the hedge fund paper, maybe post the factor model loadings for all hedge funds. That hides the proprietary raw data that you are not allowed to distribute. But you can now share the batch.py program that starts with the factor loadings and computes all results of the paper. Also, you can share programs that accesses proprietary data together with some small data sample (put a small random error on those numbers if you want to be really safe). Readers can then check that the programs are working as designed. If readers have legal access to the same database (say, have a WRDS account or bought the hedge fund data themselves), then they can even run your programs on their own data.

Writing

Word or Tex? Word tends to be simpler, and you see the final version of your text as you type. Equations are by now rather comfortable and easy to read. Automatic numbering of equations, sections, and references is still difficult. Also, one needs a separate bibliography software (Mendeley) to generate a set of references. For simply texts (like this one), Word is a good choice. Tex is more complicated, yet deals with automatic numbering and bibliography very well. You type via an editor in a messy text that contains all kinds of formatting commands in addition to your text. Tex needs a compilation step to show the final print version. The commercial Overleaf software is a cloud-based Tex editor that allows coauthors to collaborate on a paper. Unfortunately, Overleaf still requires cumbersome editing and compiling.

An interesting intermediate choice is Lyx as it is Tex-based but has an almost what-you-see-is-what-you-get interface. So one can comfortably edit a document that largely looks like the final print version, yet one has all the advantages of Tex. Lyx also allows for comments to coauthors, which is more cumbersome in Tex.

Keep your paper as part of the programs under the control of a versioning software (Git).

All programs have spellcheckers. Use them, as typos suggest that you do not care much about the paper. Why should the referee then care? Spellcheckers are not perfect ("form" and "from" both seem correct to the spellchecker, but you probably wanted the latter). Grammar checkers do not work.

Help on Writing

Below, I will repeat observations from Cochrane (2005), Zinsser (2012), and other authors. Zinser (2012) is the 30th anniversary edition, so some of the material is a little dated. A very short and even older classic is Strunk and White (1920). For a thorough study of grammar and all other aspects of writing, I suggest Hacker (2009) or the shorter Casagrande (2018). Lamott (1994) and Truss (2005) are shorter on content and longer on anecdotes.

Two important distinctions should be made. First, a research paper uses written speech, not vernacular speech (i.e., spoken words or dialects). Thus, colloquial terms (contractions such as y'all, informal words, slang) have no place in a research paper. Be careful in using idioms too (keeping up with the Joneses, selling like hot cakes). The meaning of an idiom cannot be understood solely from the meanings of the separate words composing the idiom; that is, it is not translatable. Their use should add meaning and is likely context dependent. "Keeping up with the

Joneses" could be the most appropriate expression in a paper discussing household spending. However, "hedge funds selling like hot cakes" is better phrased as "hedge funds are in high demand." Along with idioms, avoid common metaphors, similes, and figures. A hedge fund does not ride into the sunset but shuts down or delists. And should rhetorical questions be part of the paper? No, they should not.

Second, grammar is about the rules of the English language. Style is about the use of the language. A research paper ought to be grammatically correct, but its style depends on the journal's style, which the journal usually enforces once your paper is accepted. In terms of developing a personal voice, authors have freedom in how to express their ideas through word choice, transitions, and how they structure their arguments, etc.

Simplify and Declutter

Reread your paper as often as you can bare and eliminate poor writing. Recall: present tense, active voice, short sentences, short paragraphs of about 100 words, few adjectives, few adverbs, little stressing, and no "5-dollar" words. Good papers are not written in one sitting but have been reworked many times. Do yourself a favor and force yourself to rewrite them as often as you can. And then another time. And one more. But know when to stop.

Always default to clarity. Academic writing is not synonymous with convoluted sentence structures. "The utilization of a multitude of techniques will optimize our capacity to navigate the challenges that must be overcome in attempting to answer the question of hedge fund performance" is a poor way to express "We apply several approaches to answer which factor model best explains hedge fund returns."

Also, surprising as it may be, you know more about your topic than anybody else. You wrote the paper and spent time on it. Except for a few experts, even a Nobel-Prize-winning colleague will not know your topic as well as you do. So you need to explain in simple words. Do not assume that everybody knows how to do hedge fund research. Explain briefly and simply what you did and why. If this gets too long, move it into (internet) appendices. Provide examples that help the reader understand what you are doing. Figures are more powerful than tables in conveying information. Imagine a figure of adjusted R-squared values for each hedge fund (organized by numerical hedge fund ID). Use stars for the standard factor model and circles for the new factor model. If circled adjusted R-squared values are always higher than starred ones, you have a neat visual claim that your new model seems to be working. Next, back it up with some statistics.

Writing Rules

Use full stops (the period at the end of a sentence) and nothing but. Exclamation marks, semicolon, dash, colon should be used only if needed. Setting things off is best done with parentheses "()".

- Avoid contractions ("do not" instead of "don't").
- Phrase positively: "Results are insignificant" instead of "results are not significant."
- Mood changers ("but", "yet", "however", "still") are useful to help the reader: "We find a positive relation, but during crises, the sign changes."
- Avoid forms of "to go", "to get", "to do", "to be (including 'is' and 'are')" and use proper verbs
- Avoid the "-ing" form of verbs. "When we extend the sample . . ." instead of "By extending the sample . . ."
- ***<u>NO SCREAMING!!!</u>***

That or which? Typically, we want to identify one particular case by a defining characteristic. Thus, use "that" and without a comma. "The hedge funds that we investigate have several features." There exist hedge funds, and we investigate those hedge funds. To check, insert "very": "The very hedge funds that we investigate . . ." If the information is purely informative but does not identify the hedge funds, then use "which" with comma: "The hedge funds, which we investigate, have several features.” There exist hedge funds (some under investigation, some not), and all have features. To check, eliminate the text within the commas and see that the sentence still makes sense: “The hedge funds have several features”.

Gendering in English is thankfully so straightforward that I avoid he/she altogether. “The manager” or “the author” works fine. If one has two actors over and over (the principal and the agent), possibly make one a woman and one a man.

No clearing your throat. Start with the contribution. If you find a paragraph announcing what is to follow, then delete it and get going.

Eliminate jargon and unnecessary words (“It can be seen that...”, “Let me conclude...”, “econometric estimation technique” vs. “regression”, “a little”, “kind of”, “rather”, “quite”, “very”), see Appendix: Unnecessary Words. Then make a full stop. Then add the next logical thought. See that it fits with what went before and what goes after. Also, eliminate whole sentences and paragraphs. Often, the paper does not need the meandering paragraph on mutual fund data; this is a hedge fund paper, I do not use mutual fund data, so out it goes. If a sentence or paragraph gives you much grief after rewriting it, try deleting it. Often it is not needed anyway.

No waffling. Find all instances leading up to a “that” (“It is commonly known that hedge funds often are not literally hedged”). Delete up to “that” and see that the sentence still works.

Note the “often” and do not delete it. Without, you claim that no hedge fund is ever hedged and some referee will be delighted to point out a counter example. Guard yourself with qualifiers (“often”) and only claim unassailable truths (“the sun is rising every day,” mathematical proofs).
Organize the writing linearly. In the introduction, cover each important item once in logical order of appearance. Then repeat the whole story (with more details) in the rest of the paper. Do not point ahead (“as we will see later”) except to briefly mention robustness issues that you detail later

(“Results are robust to eliminating the crisis period, see Section Robustness”). Try not to recall results mentioned earlier but rather discuss them comprehensively only once.

Keep footnotes and parentheses to a minimum. They break the flow of reading. If it is important, it needs to be in the text. If it is not, delete it. Any remaining items make up your footnotes. If you have a really long footnote, move that into an appendix.

Use abbreviations sparingly unless they are widely known (CAPM) or used in every second sentence (HF for hedge fund, say). Define abbreviations on first use.

Consistency

Be consistent in how you write and keep a personal style sheet, see Appendix: Sample Style Sheet. On that, write down your editorial choices (“riskfree” vs. “risk-free”), the tense (present tense), and pronouns (“I” as opposed to “we”). Then stick to it. Be consistent. If you write about firms, then call them firms and always do so. It only confuses to call them companies, entities, or objects of interest in a desire to introduce variation.

Journals have their own style sheets. Borrow from those if you like but do not rework your draft prior to each journal submission according to that journal’s style sheet. Similarly, format the literature in a consistent style. Even better, have the computer do it and use a common style file (say, *Journal of Finance*). This is good enough until the paper gets accepted, and then a managing editor will tell you what to do for publication.

Using “I” is just fine for single-authored papers. Use “we” only if you want to include the reader. Do not use passive to avoid the “I” (“It has been found . . .” – by whom?).

Objectivity vs. Shaping the Argument

You market your paper and there is no way to write a truly objective paper on any question. Each person views the answer a little different. So the task is to set out the evidence in favor of your view and let the reader decide if they follow your arguments and accept your result.

You will stress evidence consistent with your view and argue that other findings might not be that relevant. Like a lawyer, you have an agenda: convey a credible story about some economic mechanism. The reader is judge to your lawyerly efforts.

That means that it is unethical to suppress data that does not suit your agenda. But it is fine to show an effect in a subsample, point out that during another subsample the effect did not exist, and explain why the second sample is unsuitable (say, a crisis dominated the effect). The reader can then decide what to make of your argument.

I view *p*-hacking (inflating *p*-values by searching over many models) similarly. It is unethical to run 20 models and report only the single best model. But using a factor model of returns is fine as the reader understands that many other people did this before and, thus, there is common knowledge that certain factors will work.

Two thoughts on ethics. First, the *Wall Street Journal* test is a powerful device on letting you see what you knew anyway. If somebody reported your behavior on the first page of the *Wall Street Journal*, how would you feel about that? Bad? Well, then you should not do it! Second, acting in a way that you would find acceptable in other colleagues is a good choice for yourself.

Choices need to be made in writing papers. Some can be agonizing (making the paper more practitioner friendly vs. catering to an academic crowd). Do not agonize but move on. If choices are obvious as one solution is much better than the other, pick the better one. If choices are agonizing, flip a coin. Really. Agonizing means that the utility of either choice is just as high as the utility of the other. So there ought to be little regret in going with one solution or the other. Pick one, and move on.

Feedback

You have completed all sections of your manuscript. Congrats! Give yourself a break, at least 24 hours. Longer, such as a week, two weeks, whatever your schedule allows, can give you a fresh perspective. Time away from the manuscript clears your eyes in revising and editing the manuscript yourself.

If your sole goal is to catch typos, ask a really nice friend or significant other for a cursory read, then buy them a coffee. This is not the same as copy editing.

Copy editing (or proofreading) checks for logic, style, grammar, structure, and much else. Copy editing is hard work and should not be taken lightly. Ask trusted friends, colleagues, or your adviser. As good writers tend to be in much demand for such services, they might not like being asked or give only superficial feedback. If the paper is important (job market paper, submission to a top journal), consider spending money on a professional copy editor. For a few hundred dollars, you will see a noticeable improvement – even if you thought the paper was already pretty good. This very paper went through hours of hard work by Cindy Gross (http://cmgediting.com/) and much improved in the process. Copy editing cannot fix your economics, but writing and logical flow of arguments will be much smoother. Modern copy editing uses Track Changes in Word or Comments in PDFs. Handwritten copy editing marks on printed copies of the paper are standardized, see Appendix: Copy Editing Marks.

If you like input on the economics, then identify colleagues who have a genuine interest in your particular topic. This way you are much more likely to generate useful feedback. Sending it to random people to generate comments will not work. Groom colleagues to help you and help them in turn.

For visibility, have a website and put all your contact details into the first footnote of your paper. Add a date and possibly a link to your paper where the reader can find the latest version of the paper. Put the paper on SSRN, so that it gets distributed.

Presenting Papers

Once you have a paper, you might want to present it. During a presentation, time flies. So do not waste it. I budget a few minutes per slide, so 15 slides for 45 minutes of presentation is doable. 45 for 45 is not. Leave room for discussion and questions. Presenting the paper at conferences or seminars can generate comments but do not expect too much.

Follow the above format for the paper but get to the motivation and the main research question on the first slide of your presentation. Use bullet points to guide the listener, not long sentences that you read out load while the listener tunes out (or, worse, the listener reads the long text and tunes out). Then talk freely about these bullet points if you can. Otherwise, have a discrete extended bulleted version on little cards that you can scan while holding them in your hands. If you are nervous, you will start manipulating things with your hands. Try not to give your hands access to anything like a pencil, button, clothing item, or curl of hair.

Skip the literature or have it on a slide and then just mention that there is a literature and proceed to the next slide. The few minutes of the presentation are about your work and not about what everybody else did. The model/methodology comes next and needs to be explained in simple terms – many colleagues do not work in this area. They are not stupid, they just do not know the jargon and methods of your area. So explain, maybe with a simple example. Try not to get bogged down in complicated models. Highlight a base case in sufficient detail so that the listeners can follow, but skip the extensions to get to the results.

If there are early questions that go beyond a simple clarification (“Was your variable R the hedge fund return or the risk-free rate?”), deflect them and point out that you will cover this question as you proceed. Possibly jot down the point for later reference.

Data should be handled briefly. Have “skipper” slides ready (say, on filtering the data or on a derivation of some result) that you only use if somebody really wants to know some detail.
Then get to the main result as fast as possible. Avoid giving an early version of results in the introduction, as that invites a rowdy and uninformed discussion right there. Again, present one nice table/figure for the main result, which you carefully explain. Try to get to the minor results (and mention here how the base methodology needs to be changed to obtain these results) before there is a general discussion. If this does not work, then use the discussion to introduce your minor results. At this point, you achieved what you wanted in your presentation.

Essentially stop here. You can put up one more slide on robustness and mention that along all these dimensions (bullet points) your results are robust and one conclusion slide that just states in three bullet points what you did. Try to not even read that out aloud but thank the audience for their attention and ask for questions.

Listen to the questions carefully, and then wait a moment. Answer carefully and to the point. If you need more time to think, say so and take your time. Avoid shooting from the hip and then having to correct yourself. If some listener monopolizes the discussion with three follow up questions, pick one, answer that, call on the next listener. Write down interesting questions and ideas. A simple thank you is an appropriate reaction to a great idea. Do not answer very difficult questions where you do not know the answer. Thank the colleague and point out that you need to work that out. Possibly promise to get back with the result, but then you have to keep that promise. Mind you: nobody ever complained about a talk ending 5 minutes early! But your listeners resent if you overrun and steal their coffee break.

Submission to a Journal

Try to assess the quality of your paper and then overshoot only somewhat. Sending a C-level paper to the *Journal of Finance* will get you a desk rejection and editors have memory, too. Ideally, you already wrote the paper with some target quality – the more of a splash you want to make on the job market, the better the paper needs to be. The acceptance rate at journals is low (1/10 or worse at the top, not much easier further down). Turn around in finance is around 100 days, and longer in economics. So, if you like to have something published, plan ahead. The job market normally does not require a publication, but having a publication already is a strong signal.

Submit finished papers only. Sending an unfinished paper to a journal to solicit comments is a bad idea. A journal's managing editor plays gatekeeper for unfinished manuscripts. If an unfinished manuscript is somehow assigned to an editor and referee, they are likely to reject it immediately, and you lose your chance of publishing at that journal. Do not expect a referee to do your work for you and write a detailed outline of how to fix your unfinished work.

Follow the submission requirements of the journal (not the style requirements for publication, which are more onerous and come much later); journals are picky about those. They can be found on the website of each journal and tend to cover formatting (font size, line spacing, and the location of tables/figures). If the journal in question publishes accepted manuscripts, have a look at them. They should be in line with submission requirements.

A short cover letter should be sent too. I only use a longer cover letter if the paper is complicated (reject & resubmit, there was some earlier version at the journal in question, or the paper is related to a special issue of the journal).

Revise & Resubmit and Rejections

In case of a revise and resubmit, do the requested work unless it is patently absurd (in that case, contact the editor). Write a detailed letter to the editor, copying the original requests into the text, explaining how you dealt with the requests (use different fonts for requests and reply – bold italic vs. normal). Same for the associate editor and the referees. Not all changes need to make it into the paper, but they all need to be mentioned in the letters. Chances of acceptance are now about 1/2, so do not blow this chance.

I treat reject and resubmit in a similar way to revise and resubmit, but with more extensive rework.

Upon a flat-out rejection, you should clearly address outright shortcomings. Fix what needs fixing.

That said, many rejections are uninformative ("The paper is interesting but not sufficiently so") or unhelpful ("The paper is interesting but one should rather write a paper on this completely different topic"). At that point, righteous indignation could inspire you to open a formal complaint with the editor. Do not do so. It is expensive, normally not leading to the desired result, and irritates the editor. Sleep over it. Read it again, and ask yourself if maybe the paper was not that great after all. Only if the referee is outright wrong ("This author does not take logs of returns, which would change all results. Thus, I reject the paper.") entertain a complaint to the editor.

As one cannot even address the concerns due to the rejection, the best strategy is often to fix any obvious shortcomings and then submit to different journal of same quality or lower. Stop at some level of journal quality and consider making the paper a permanent working paper. You can publish anything at an E-level journal, but it also scars your curriculum vitae. If you want to signal quality research, then do not publish too low.

This strategy of gradually reducing journal quality is wasteful in terms of scarce refereeing capacity, but optimal from the author's perspective. Given the noise in refereeing, the next referee might come up with very different comments. If it goes to the same referee again at the new journal, that referee should excuse themselves as to not create double jeopardy for the paper.

Refereeing

Once you submit a paper for publication, journals will take note of you and ask you to referee other papers. If you have absolutely no time or a conflict of interest, say so now. If you want to publish in that journal yourself, try to find the time for writing a good referee report. If you would never publish in that journal, you might want to forgo the assignment. Writing a good report easily takes a day or longer. A good starting point in deciding how to write a referee report is to review a collection of referee reports from some trusted senior colleague.

A referee report has two parts: a personal communication with the editor and the referee's report proper. In the personal communication you write a suggestion (please reject this paper) and any private information that should not be shared with the authors (conflict of interest, acknowledgment of own work competing with the paper in question, the suggestion that another expert in some other area needs to assess certain aspects, etc.). Do not repeat information from the referee report.

The referee report should start with the title of the paper and a summary in your own words of the main contribution. This serves to make sure that you understand the paper and that the paper is sufficiently coherent. If the paper is atrocious, then a short statement is all that is needed.

Otherwise, in order of occurrence in the text, one should list the Main Comments (numbered). These are issues that need to be addressed for the paper to proceed. You might question the relevance of the research question (too narrow). Or the far-reaching assumptions required in the model. Or the statistical problems that riddle the empirical work. However, these comments should not reflect your personal taste but only point out problems severe enough to question the validity of the paper. Based on these comments, form your recommendation to the editor: reject; reject and resubmit a completely new version of the paper; revise and resubmit; conditionally accept subject to some additional test or writing; or accept. If the paper has severe enough problems that fixing all of them seems a daunting task, rather reject than invite a resubmission.

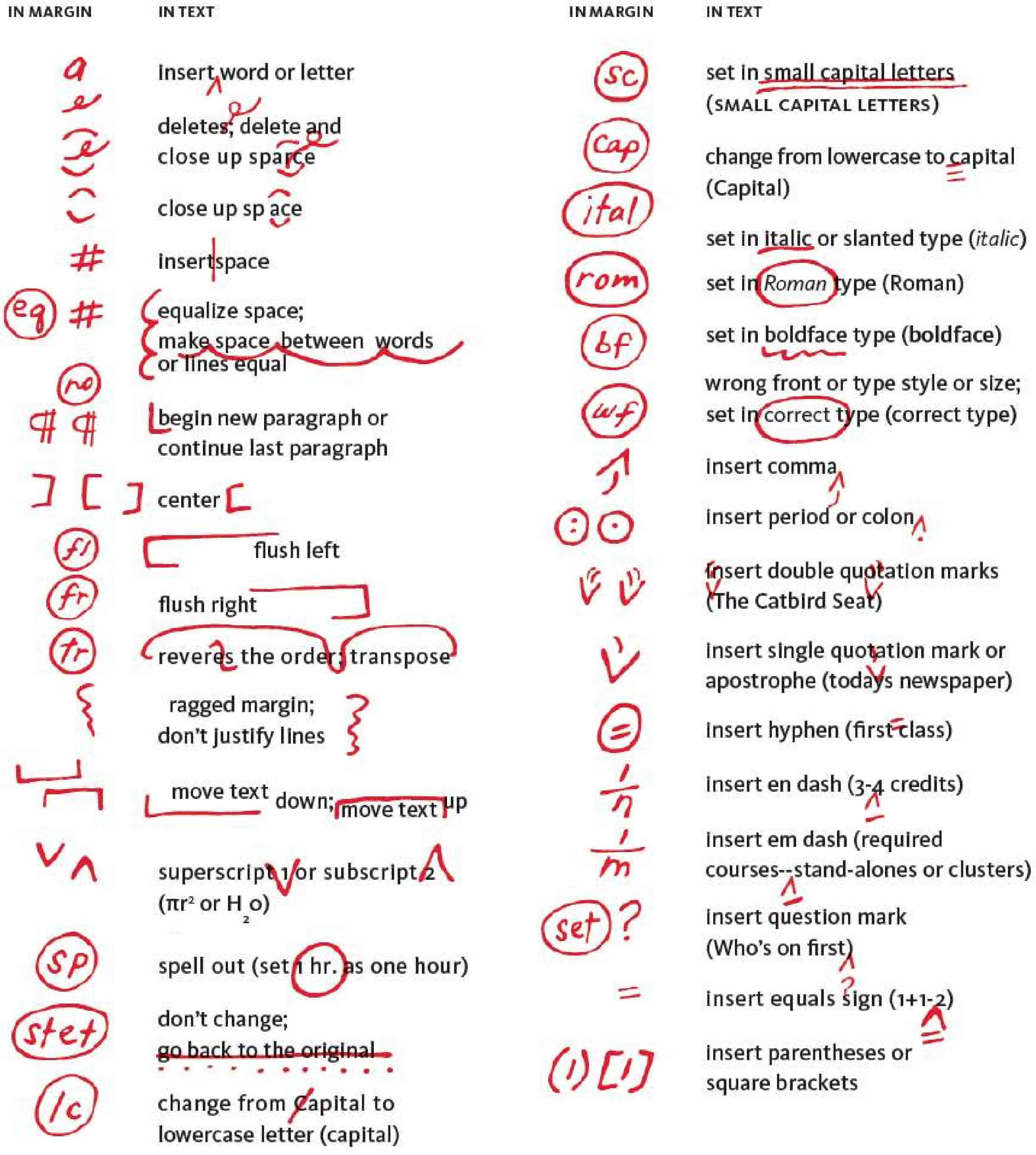
The final part of the report is Detailed Comments (numbered again and ideally with page and paragraph references to where the problem occurred). Here, go with low-level issues, such as typos, suggestions, alternatives, extensions, requests for clarification, and formatting issues. These problems would not affect the publication decision but might be helpful for the authors. The tone of the referee report should be civil. There is no reason to be rude, and you can make a strong point in a polite way.

Conclusion

I hope that these thoughts were of some use for the beginning PhD student. Much of the advice has been common sense, but maybe one tip or another can smooth your entry into the profession. Please distribute this document freely, and send me feedback or questions to jens.jackwerth@uni-konstanz.de.

After learning and adhering to everything I have said, start violating every rule and piece of advice. With experience, you should appreciate the rules and understand why I suggested them. But mastery implies that you can and should go against the rules as you see fit. Good luck!

Appendix: Copy Editing Marks



IN MARGIN	IN TEXT
a	insert word or letter
	deletes; delete and
	close up sparce
	close up sp ace
#	insertspace
eq #	equalize space; make space between words or lines equal
no ¶ ¶	begin new paragraph or continue last paragraph
] [ ]	center
fl	flush left
fr	flush right
tr	reveres the order; transpose
	ragged margin; don't justify lines
	move text down; move text up
V Λ	superscript 1 or subscript 2 (πr^2 or H_2o)
sp	spell out (set 1 hr. as one hour)
stet	don't change; ~~go back to the original~~
lc	change from Capital to lowercase letter (capital)

IN MARGIN	IN TEXT
sc	set in small capital letters (SMALL CAPITAL LETTERS)
cap	change from lowercase to capital (Capital)
ital	set in italic or slanted type (*italic*)
rom	set in *Roman* type (Roman)
bf	set in boldface type (**boldface**)
wf	wrong front or type style or size; set in correct type (correct type)
,	insert comma
⊙ :	insert period or colon
“ ”	insert double quotation marks (The Catbird Seat)
’	insert single quotation mark or apostrophe (todays newspaper)
=	insert hyphen (first class)
1/n	insert en dash (3-4 credits)
1/m	insert em dash (required courses--stand-alones or clusters)
set ?	insert question mark (Who's on first)
=	insert equals sign (1+1-2)
(/) [/]	insert parentheses or square brackets

Appendix: Unnecessary Words

This appendix suggests words that can be eliminated. It also lists “5-dollar” words – and simpler replacements.

analyze – discuss

and/or – or

comprise

concept – idea/notion/thought

critique – criticize

diligently

due to – because

during which – when

even

finalize

for convenience

former … latter

fortunately

hence

hopefully

however

hypothesize – suspect

implement

in order to – to

in other words

in terms of

interesting

into – in

intra/inter – within/between

is equal to – equals

kind of

obviously/of course

respectively

solely – only

sort of

state – say

structure

the above/the preceding

the paper – our work

the process of

this/that/those/these – this hedge fund/that manager/those regressions/these principals

thus – thereby

time frame – time

type of

very

via – by

vis a vis

whether or not – whether

within – in

Appendix: Sample Style Sheet

Use I instead of we

Present tense but past tense if obviously in the past

riskfree

In Word: Alt-0150 N-dash – ; Alt-0151 M-dash —

February 21, 2020, through February 28, 2020, . . . OR February 21, 2020 – February 28, 2020.

hyphenate: high-volatility, closed-form, shorter-maturity, 1.0-year, downward-sloping, tick-level, five-year, 24-month

“. . . deliver higher Sharpe ratios than does the market” needs the “does” after “than”

toward (US, towards is UK)

“only” always after the verb

…; see Jackworth (2000).

… (see, e.g., Jackwerth, 2000).

Times get shorter/longer, quantities get smaller/larger

“where” refers to a place – “in which” refers to a model or formula

“data” is always plural – use only for data

Never point ahead (“below we show”) and rarely point back (“as stated above”).

“Figure 5 depicts . . .” “Table 1 shows . . .” “Figure 7 confirms . . .” NOT “We show in Figure 5”

In labels of Figures/Tables, start with: “Prices of . . .” NOT “We depict the prices of . . .“

Panels need a header next to the label: “Panel A. Returns on . . .” No list of panels in the header

Variables in italics, but not numbers: $X = 0$.

“Upon revision the main part is . . .” NOT “Upon revision, the main part . . .” No comma needed if the new phrase (starting with “the”) is clear

References

Casagrande, June, 2018, The Joy of Syntax: A Simple Guide to All the Grammar You Know You Should Know, Ten Speed Press, Berkeley.

Cochrane, John, 2005, Writing Tips for PhD Students, *Working Paper, University of Chicago*.

Genztkow, Matthew, and Jesse M. Shapiro, 2014, Code and Data for the Social Sciences: A Practitioner's Guide, *Working Paper, University of Chicago.*

Fung, William, and David A Hsieh, 2004, Hedge Fund Benchmarks: A Risk-Based Approach, *Financial Analysts Journal* 60, 65–80.

Hacker, Diana, 2009, Rules for Writers, 6th edition, Bedford/St. Martin's, Boston.

Lamott, Anne, 1994, Bird by Bird, Anchor Books, New York.

Rampini, Adriano, S. Viswanathan, and Guillaume Vuillemey, 2021, Retracted: Risk Management in Financial Institutions, *Journal of Finance* 76, No. 5, 2709.

Skrainka, Benjamin S., 2010, Enhance Your Productivity and Software Quality with Techniques from Silicon Valley, Presentation Slides, University College London.

Strunk, W., Jr., and E.B. White, 1920, The Elements of Style, Harcourt, Brace and Howe, New York.

Truss, Lynne, 2005, Eats, Shoots & Leaves, Gotham Books, New York.

Zinsser, William, 2012, On Writing Well: The Classic Guide to Writing Nonfiction, 7th edition, Harper Collins, New York.