Data Appendix

Infrequent Random Portfolio Decisions in an Open Economy Model

Philippe Bacchetta, Eric van Wincoop, and Eric Young

February 2022

A. Data Availability Statement

Portfolio data from Bertaut and Tryon (2007) can be downloaded as Excel files from https://www.federalreserve.gov/pubs/ifdp/2007/910/default.htm.

Data from Bertaut and Judson (2014) can also be downloaded as Excel files from https://www.federalreserve.gov/econres/ifdp/estimating-us-cross-border-securities-positions-new-data-and-new-methods.htm.

MSCI data can be accessed via a subscription through a financial data provider, e.g. Datastream.

B. MSCI Data

Using MSCI data notation, the precise data definition for excess returns is:

$$\begin{aligned} er_t &= \ln\left(\text{msci\_us\_TR}_t/\text{msci\_us\_TR}_{t-1}\right) \\ &\quad - \ln\left(\text{msci\_acwi\_exus\_TR}_t/\text{msci\_acwi\_exus\_TR}_{t-1}\right) \end{aligned}$$

where msci_us_TR is "MSCI US Total Return Index" and msci_acwi_exus_TR is "MSCI ACWI ex US Total Return Index" (ACWI stands for "All Country World").

d_t^D is computed as relative earnings and earnings are derived by dividing the price index (PI) by the price earnings ratio (PER):

$$d_t^D = \ln\left(\text{msci\_us\_PI}/\text{msci\_us\_PER}\right) - \ln\left(\text{msci\_acwi\_exus\_PI}/\text{msci\_acwi\_exus\_PER}\right)$$

q_t^D is computed as relative log price index: $q_t^D = \ln\left(\text{msci\_us\_PI}/\text{msci\_acwi\_exus\_PI}\right)$. The moments in Table 3 are computed for the interval November 1995 - December 2018.

C. Portfolio Data

Portfolio shares are defined in Section 4.1. US market capitalization: msci_us_MV; ROW market capitalization: msci_acwi_exus_MV.

US external claims on ROW : us_stk_est_pos from bertaut_tryon_claims_thru2011.csv and bertaut_judson_positions_claims_2019.csv. ROW external claims on US: ftot_stk_est_pos from ticdata.liabilities.ftot.txt and bertaut_judson_positions_liabs_2019.csv. Both are for all countries, item 69995. We use all countries for ROW rather than using bilateral data for the 44 countries in MSCI data. Bilateral country data may be biased because it does not always capture the true destination or source country (e.g., portfolios with financial centers).

D. Excess Return Predictability, Momentum, and Reversal

We use MSCI data, but extend the sample to 73 countries for 1970:01 - 2019:02. The countries are argentina, australia, austria, bahrain, bangladesh, belgium, bosnia herzegovina, brazil, bulgaria, canada, chile, china, colombia, croatia, czech republic, denmark, egypt, estonia, finland, france, germany, ghana, greece, hong kong, hungary, india, indonesia, ireland, israel, italy, jamaica, japan, jordan, kazakhstan, kenya, south korea, kuwait, lebanon, lithuania, malaysia, mauritius, mexico, morocco, netherlands, new zealand, nigeria, norway, oman, pakistan, peru, philippines, poland, portugal, qatar, romania, russian federation, serbia, singapore, slovenia, south africa, spain, sri lanka, sweden, switzerland, taiwan, thailand, tunisia, turkey, ukraine, united arab emirates, united kingdom, vietnam, and zimbabwe.

The estimates in Table 4 are computed using a panel with country-level variables relative to the US. Data is incomplete for several countries, so that we have an unbalanced panel. All regressions include a country fixed effect and standard errors are clustered by month.

E. Computer Program Instructions

The main executable file is bvwy_final_restud.f90. It must be linked with the following auxiliary files to compile: hbrd.f90, helph36.f, matrix4.f, random2.f, and roots.f90. The code was compiled using an Intel Compiler linked through Microsoft Visual Studio with optimization flag -O3; although it should work with other compilers there is no guarantee.

Note that the code is only guaranteed to work at the specified parameter values; changing any parameter values, even by seemingly-trivial magnitudes, frequently causes the code to fail. The specific program solves the model for $p = 0.1$; since the model does not solve quickly or robustly, the files include hot-start input files, without which the code may or may not run. Hot-start input files for alternative values of p are included in the replication package. The case of $p = 1$ is very sensitive and is solved iteratively starting at $p = 0.999$ and progressively increasing p (which is done by hand). While the $p = 1$ case technically has fewer state variables, the global solution for the simpler state space is not well-behaved, so the model is solved as if these other state variables are relevant. The case with $p = 1$ and small financial shocks is even more ill-behaved and is analyzed using only the local solution. Runtimes are very long, often several days, especially if you wish to compute the portfolio approximation terms d and D.

There are three modules at the top of the main program file; parameters_mod contains variables that are fixed for the entire program (parameters, integration nodes), states_mod contains variables, like state vectors, that must be shared with the nonlinear equation solver, and special_mod contains variables that must be shared with the root finder that computes relative wealth w_d. Interfaces (defining the subroutines used in the solvers) can be found at the beginning of the relevant parts of the program.

The main program first sets of the integration nodes, which follow a 33-point monomial rule (Judd 1998), translated to allow for correlation between shocks. Svbar contains the steady state values of the state variables (there are 15, but each household only needs to know 12 of them and aggregates depend only on the first 9). The code then solves the Euler equations in the appendix at the steady state, reading an initial guess from the xbest_p10.txt file; this solution will be referred to as the local solution. The code then reads in the nodes for the global solution from the file clusters_new4.txt, which were obtained by simulating the local solution and running the Matlab code bvwy_cluster.m (which is the only non-Fortran code used); uncommenting line 382 runs the subroutine simulation0, which constructs the simulation that is used to construct these points using Ward's clustering algorithm. The file xx168.in contains the existing solution at each of the 151 points for the global solution. Comment out line 427 (goto 6) if you want to recompute the solutions; this process can take several days and in general cannot be parallelized (for reasons I will outline below). All new solutions (whether recomputed or not) are recorded in the file xx168.out. After solving for the solutions, the code constructs the solutions at symmetric points (by switching dH and dF and tauH and tauF, taking the negative of wd, and making various other adjustments to the other state variables, we can get the solution at an additional 150 points without recalculating them). The result are local solutions at (up to) 301 distinct points in the state space. For $p = 0.04$, we are able to obtain only 143 points, but we could not detect any systematic pattern in the points at which solutions could not be obtained.

Various simulation options are available – one long simulation used for the predictability regressions in Table 4 (simulation2), multiple short simulations used for Table 3 (simulation3), impulse responses (sim_impulse) used for Figures 1 and 2, and simulations with only certain shocks active (obtained by manipulating the error terms in each simulation subroutine). Simulation9 computes the d and D terms used in the approximate portfolio expression by taking a simulation from simulation3 and running "capi forward simulations" from each point, then averaging to get expected moments. If memory is a concern, commenting out the simulation subroutines may help. The impulse responses are computed using the local solution.

The solution method at a given point is now detailed. At point s, the code constructs a system of 168 equations in 168 unknowns, which are the constants and slope coefficients of 15 functions detailed in the appendix. The equations are the market clearing conditions for stocks and bonds, the Euler equations, the envelope conditions, and the Bellman equations, along with their first derivatives (evaluated using a centered two-point stencil with step sizes that differ according to the state variable, eps for most variables and eps2 for tauH and tauF). The subroutines are fcn and fcn2 (which handles the local solution only at the deterministic steady state) and fcn3 and fcn4 (which do the global solution at arbitrary points). The first subroutine in each group calls the second to evaluate the Euler equations and the numerical derivatives, with the structural equations located in fcn2 and fcn4.

Solutions to these equations at points other than the deterministic steady state are often difficult to obtain. To find them, we initialize the search at svbar, then use a homotopy (a path of 1001 points, denoted lamb2, that begins at svbar and ends at s(t), the state point under consideration) to gradually creep up on the solution. At each step, we first use a heavily-dampened quasi-Newton solver (where the update step is 0.0025 times the Newton step) until the equations are satisfied within a tolerance of 1e-03, then switch to a hybrid-Powell method with numerical derivatives to polish the solution to a tolerance of 1e-12. If this procedure fails, which it sometimes does, we record this failure and move to the next point. The procedure fails if, at any point along the homotopy path, we encounter NaN (Not a Number) outcomes, which can happen for a variety of reasons; we do not require that the solver get an accurate solution at each point along the path, as sometimes it "finds its way back". We also record a failure if the solution at the terminal point of the path – the current state – is not within the tolerance for the hybrid-Powell solver. Note that the value function terms are normalized by their steady state level (fbar) to keep scaling problems from appearing, as the true value functions can have very large values. Expectations are computed using the monomial rule referenced above.

After attempting a solution at every point, we revisit each failure point. We locate the closest node to the failed state at which we successfully found a solution and initialize the same procedure, but starting from that point instead of the steady state. This process may need to be repeated multiple times to get solutions at all points, and for some parameter configurations we were never able to get solutions at all points (especially for low values of p). We record the number of solved points as the parameter nstar and the location of failed points in ifail (=1 if the point was not solved). The reason that this process is so tedious is that the points often cannot be easily computed in parallel, since we need good initial guesses from the solutions at nearby points, and the cluster algorithm has no way of "ordering" the points in a systematic way.

To construct a global solution, we use a modified Shepard's inverse-weighting scheme, which weights the solution at each point by the inverse of the distance between the current point and the node at which that solution applies, truncated at some maximum distance (called radius in the code and set to 4 after some experimentation). The code automatically checks whether a given point has a valid solution before constructing the Shepard weights, and it ignores any points where ifail=1. Weights are automatically renormalized to sum to 1.

To simulate, we have to solve endogenously for w_{dt}, the log of relative wealth today, since it depends on the current values of the asset prices. We consider the law of motion for w_{dt} as a nonlinear equation, which we solve using Brent's method; for the local solution this solution can be done analytically because it is linear, and our nonlinear solver finds the same solution in that case. The function for this procedure is getwd, getwd2, getwde, or getwde2, depending on whether we use the simulation (the difference is in the use of the global or local solutions) and whether we are computed the current wd or the value of w_d tomorrow in different states (used in calculations for expected excess returns).

Output files – the moments for Table 3 are stored in the file moments.out, in the order presented in the text with different sheets corresponding to different values of p; the moments for Table 4 are in regressions.out (except for the theoretical moments, which are derived from the structural parameters); the impulse response functions are in impulse*.out, where *=1 for dD shocks and *=2 for tauD shocks; the time series for the portfolio approximation (Figure A1) is in zeatilde_global.txt, column 11; the distribution of w_d (Figure A2) is in wd.out. Note that, especially for Table 3, the results may vary due to small sample size and different random numbers, as some of the moments have large standard deviations and can vary over wide ranges. Increasing the number of simulations (capn in simulation3) can help stabilize the means, but at substantial computational cost.

1 Replicate Table 4 Excess Return Predictability

1.1 Panel 1 and 2 Table 4

1.2 Panel Expected Excess Return Table 4

Regress (1) and denote the fitted values by $\hat{er}_{n,t,t+1}$.

$$er_{n,t,t+1} = a_n + a_1 er_{n,t} + a_2(d^D_{n,t} - q^D_{n,t}) \tag{1}$$

where a_n is a country fixed effect.

Table 1: Predictability International Equity Return Differentials

	(1)	(2)	(3)	(4)
	1 month	3 months	12 months	48 months
Earning-Price	0.00660***	0.0185***	0.0779***	0.490***
	(0.00197)	(0.00383)	(0.00895)	(0.0444)
Constant	-0.00177	-0.00366	-0.00849	-0.0283
	(0.00149)	(0.00294)	(0.00765)	(0.0315)
Observations	22033	21889	21241	18643
R^2	0.006	0.015	0.044	0.128

Standard errors clustered by month in parentheses. * $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$.
Notes: Results are based on panel regressions of equity returns in 73 countries minus the equity return in the US on the log relative earnings-price ratio over the period 1970:01-2019:02. All regressions include a country fixed effect.

Table 2: Momentum and Reversal

	(1)	(2)	(3)
	$q^D_{t+1} - q^D_t$	$q^D_{t+3} - q^D_t$	$q^D_{t+12} - q^D_t$
$q^D_t - q^D_{t-1}$	0.0446**	0.1020***	0.3333***
	(0.0191)	(0.0296)	(0.0622)
q^D_t	-0.0088***	-0.0290***	-0.1365***
	(0.0022)	(0.0045)	(0.0090)
Observations	20808	20712	20280
R^2	0.007	0.020	0.085

Standard errors clustered by month in parentheses. * $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$.
Notes: Results are based on panel regressions with 73 countries over the period 1970:01-2019:02. All regressions include a country fixed effect.

By country, calculate the standard deviation and the autocorrelation of $\hat{er}_{n,t,t+1}$. Then, calculate the average standard deviation and autocorrelation across countries.

Table 3 shows the coefficients a_1 and a_2 of equation (1).

Table 3: Predictability International Equity Return Differentials

	(1) 1 month	(2) 3 months	(3) 12 months
Momentum	0.0426^{**}	0.0772^{***}	0.297^{***}
	(0.0177)	(0.0296)	(0.0683)
Earning-Price	0.00716^{***}	0.0195^{***}	0.0818^{***}
	(0.00196)	(0.00381)	(0.00888)
Constant	-0.00189	-0.00384	-0.00916
	(0.00148)	(0.00292)	(0.00759)
Observations	22021	21877	21229
R^2	0.007	0.017	0.049

Standard errors clustered by month in parentheses. * $p < 0.10$, ** $p < 0.05$, *** $p < 0.01$.
Notes: Results are based on panel regressions of equity returns in 73 countries minus the equity return in the US on the log relative earnings-price ratio over the period 1970:01-2019:02. All regressions include a country fixed effect.

2 Simulation

Start by estimating equation (1) using pooled OLS. Include a country fixed effect a_n and cluster standard errors by months. Denote the standard error of the estimated a_1 and a_2 by $SE(a_1)$ and $SE(a_2)$, respectively.

Using those estimates, draw $a_1^{(j)}$ and $a_2^{(j)}$ from the normal distribution $\mathcal{N}(a_1, SE(a_1))$ and $\mathcal{N}(a_2, SE(a_2))$, respectively. The expected excess return at draw (j) is then equal to

$$E_t(er_{n,t,t+1})^{(j)} = a_1^{(j)} * er_{n,t} + a_2^{(j)} * (d_{n,t}^D - q_{n,t}^D) \tag{2}$$

For each country-iteration pair, calculate the autocorrelation (AC) and the standard deviation (SD) of $E_t(er_{n,t,t+1})^{(j)}$

$$AC_n^{(j)} = AC(E_t(er_{n,t,t+1})^{(j)}) \tag{3}$$

$$SD_n^{(j)} = SD(E_t(er_{n,t,t+1})^{(j)}) \tag{4}$$

For each country, calculate the standard deviation of the autocorrelation and the standard deviation of the standard deviation across the iterations

$$SD(AC)_n = SD(AC_n^{(j)}) \tag{5}$$

$$SD(SD)_n = SD(SD_n^{(j)}) \tag{6}$$

With J = 1000, the average value of $SD(AC)_n$ is .202 and the average value of $SD(SD)_n$ is .0011.