Markdown - Breaking in or Breaking Through? How Local Specialisations Shape the Integration of AI Technologies

# 1. Technological Spaces based on Technological field

## 1.1. Calculate Specializations for Different Time intervals

In this first part, we load the very large datasets containing patent data and inventor’s location, and use them to calculate specialisations and ultimately create the Global technological space (GTS) and the AI-specific technological space (ATS). We calculate specialisations per interval.

We start by defining custom functions and loading the first part of the large patent dataset with inventors’ location.

The patent file with inventors’ location looks like this:

head(ipc\_all\_patents\_part2\_df)

## appln\_id ctry\_code techn\_field\_nr weight priority\_year  
## <int> <char> <int> <char> <int>  
## 1: 203438 JP 2 9 2000  
## 2: 203438 JP 2 9 2000  
## 3: 203438 JP 9 1 2000  
## 4: 203438 JP 9 1 2000  
## 5: 203521 US 15 375 1996  
## 6: 203521 US 16 625 1996

The weight column was downloaded from patstat, but it it’s totally irrelevant and it will be calculated again later. We load next the data on AI patents (file “other\_files/IPCs\_AI.csv”). It looks like this:

head(ai\_patents\_df)

## appln\_id appln\_id2 patent\_office priority\_year ctry\_code source kinds  
## <int> <int> <char> <int> <char> <int> <char>  
## 1: 475222998 475222998 CN 2016 AI\_pat 7 single  
## 2: 475222998 475222998 CN 2016 AI\_pat 7 single  
## 3: 475222998 475222998 CN 2016 AI\_pat 7 single  
## 4: 475222998 475222998 CN 2016 AI\_pat 7 single  
## 5: 475222998 475222998 CN 2016 AI\_pat 7 single  
## 6: 475222998 475222998 CN 2016 AI\_pat 7 single  
## appln\_id3 ipc\_class\_symbol ipc\_class\_level ipc\_version ipc\_value  
## <int> <char> <char> <char> <char>  
## 1: 475222998 G06F 17/22 A 01/01/2006 I  
## 2: 475222998 G06F 17/24 A 01/01/2006 I  
## 3: 475222998 G06F 17/27 A 01/01/2006 I  
## 4: 475222998 G06F 17/30 A 01/01/2006 I  
## 5: 475222998 G06F 19/00 A 01/01/2011 I  
## 6: 475222998 G06N 3/08 A 01/01/2006 I  
## ipc\_position ipc\_gener\_auth  
## <char> <char>  
## 1: L CN  
## 2: L CN  
## 3: L CN  
## 4: F CN  
## 5: L CN  
## 6: L CN

We also load the IPC file with information about technological fields (file “other\_files/ipc\_technology.csv”), which after processing looks like this:

head(ipc\_names\_df)

## field\_nr sector field\_name  
## 1 1 Electrical engineering Electrical machinery, apparatus, energy  
## 2 2 Electrical engineering Audio-visual technology  
## 3 3 Electrical engineering Telecommunications  
## 4 4 Electrical engineering Digital communication  
## 5 5 Electrical engineering Basic communication processes  
## 6 6 Electrical engineering Computer technology  
## techn\_field\_nr  
## 1 1  
## 2 2  
## 3 3  
## 4 4  
## 5 5  
## 6 6

Now we’ll start calculating the specialisations of countries and AI technologies per interval. We have three intervals, the first being from 1974 to 1988, the second from 1989 to 2003, and the third from 2004 to 2018. We start by breaking the large patent file into it’s corresponding interval, and applying the two custom functions over it (group\_by\_applnID and group\_by\_ctry\_and\_techn\_field).

The first function (group\_by\_applnID) groups data by application ID, and within each group, calculates a field-specific weight equal to 1 divided by the number of technological fields in that group. After applying this function over our interval-specific patent data, it looks like this:

head(region\_tech\_fields\_1\_df)

## # A tibble: 6 × 4  
## appln\_id ctry\_code techn\_field\_nr field\_weight  
## <int> <chr> <int> <dbl>  
## 1 206163 DE 1 0.5   
## 2 206163 DE 1 0.5   
## 3 214019 FR 9 0.25  
## 4 214019 FR 9 0.25  
## 5 214019 FR 29 0.25  
## 6 214019 FR 29 0.25

Next we apply the second function (group\_by\_ctry\_and\_techn\_field), which aggregates the weighted fields at the country-technology field level. After applying it, the data looks like this:

head(region\_tech\_fields\_1\_df)

## # A tibble: 6 × 3  
## ctry\_code techn\_field\_nr n\_tech\_reg  
## <chr> <int> <dbl>  
## 1 AD 20 1  
## 2 AD 24 1  
## 3 AD 28 3  
## 4 AD 32 1  
## 5 AD 33 1  
## 6 AD 34 3

This file is saved as a csv for being used later (for this first interval, the name and location of the file is “Files\_created\_with\_the\_code/data/files\_code\_Fields\_analysis/reg\_tech\_FirstPeriod.csv”). We use this file in the next calculation, in which we create a matrix from this aggregated data. The matrix counts the name of registers of each country in which possible technological field. It looks like this:

print(as.matrix(mat\_reg\_tech1[1:20, 1:12]))

## 1 2 3 4 5 6 7 8 9 10 11 12  
## AG 1 0 0 0 0 0 0 0 0 0 0 0  
## AM 1 0 0 0 0 0 0 0 0 0 0 0  
## AR 11 7 6 1 2 2 0 0 3 12 1 6  
## AT 758 299 198 23 131 64 1 28 279 461 38 130  
## AU 1403 606 409 48 178 272 10 81 509 1321 149 659  
## BA 0 0 0 0 0 0 0 0 0 0 0 0  
## BB 0 0 0 0 0 0 0 0 0 1 0 0  
## BE 261 97 159 32 54 43 0 26 182 196 65 80  
## BG 992 335 166 36 333 455 0 114 228 1284 137 340  
## BI 1 0 0 0 0 0 0 0 0 0 0 0  
## BM 2 2 0 0 0 0 0 0 0 0 0 0  
## BO 2 0 0 0 0 0 0 0 0 1 0 1  
## BR 1759 647 754 47 138 325 12 50 296 962 48 851  
## BS 4 0 0 0 0 0 0 0 0 0 0 0  
## BU 0 0 0 0 0 0 0 0 0 1 0 0  
## CA 3827 1409 1474 254 553 784 14 372 1368 2915 252 1041  
## CH 2194 788 372 108 299 242 4 225 841 2526 120 754  
## CL 0 1 1 0 0 1 0 0 1 1 0 2  
## CN 931 233 146 20 106 317 0 132 306 994 45 223  
## CO 8 2 0 0 0 0 0 0 0 0 0 4

We finally use this matrix to calculate the general specialisations (RTA indexes, which stands for Revealed Technological Advantage) of countries in this first interval. The RTA is set to be non-binary, and the data looks like this after this calculation:

head(reg\_RCA1\_df)

## # A tibble: 6 × 3  
## ctry\_code techn\_field\_nr RCA  
## <chr> <chr> <dbl>  
## 1 AG 1 6.41   
## 2 AM 1 12.8   
## 3 AR 1 0.421  
## 4 AT 1 0.748  
## 5 AU 1 0.479  
## 6 BA 1 0

We do the same for the specialisations of countries in AI considering the AI patents in this interval. The steps are pretty much the same: we separate the AI data into this specific interval (ai\_patents\_df), apply the two custom functions (group\_by\_applnID and group\_by\_ctry\_and\_techn\_field), save the file (“Files\_created\_with\_the\_code/data/files\_code\_Fields\_analysis/Data1period\_RCA\_techn\_field.csv”), create a matrix, and use it to calculate countries’ specialisations in AI for this period. The AI-related specialisations data looks like this:

head(reg\_RCA1\_AI\_df)

## # A tibble: 6 × 3  
## ctry\_code techn\_field\_nr RCA  
## <chr> <chr> <dbl>  
## 1 AT 1 0  
## 2 AT 10 0  
## 3 AT 11 0  
## 4 AT 12 0  
## 5 AT 13 0  
## 6 AT 17 0

We merge this data with the “general” specialisations calculated earlier. The resulting file and an additional example for the case of Japan look like this:

#Resulting file:  
head(rca\_data\_period\_1\_df)

## ctry\_code techn\_field\_nr RCA\_Gen RCA\_AI Period  
## 1 AD 1 0 NA 1974-1988  
## 2 AD 10 0 NA 1974-1988  
## 3 AD 11 0 NA 1974-1988  
## 4 AD 12 0 NA 1974-1988  
## 5 AD 13 0 NA 1974-1988  
## 6 AD 14 0 NA 1974-1988

#Example Japan:  
head(rca\_data\_period\_1\_df[rca\_data\_period\_1\_df$ctry\_code == "JP",])

## ctry\_code techn\_field\_nr RCA\_Gen RCA\_AI Period  
## 2731 JP 1 1.1268685 1.4025974 1974-1988  
## 2732 JP 10 1.0109760 1.2022263 1974-1988  
## 2733 JP 11 0.7067914 0.0000000 1974-1988  
## 2734 JP 12 1.0644547 1.0957792 1974-1988  
## 2735 JP 13 0.6104061 0.7012987 1974-1988  
## 2736 JP 14 0.6177233 NA 1974-1988

We save this data for later usage (“Files\_created\_with\_the\_code/data/files\_code\_Fields\_analysis/Data1period\_RCA\_techn\_field.csv”). Next, we turn to the AI-specific specialisations of this interval. We pick our AI data for this interval and replace their country codes by a new “AI-specific” fake code named AI\_pat. This allows us to calculate specialisations for AI as it were a country exploring distinct technologies. We apply the same two custom functions over it (group\_by\_applnID and group\_by\_ctry\_and\_techn\_field), and get the following resulting file with the AI specialisations:

head(region\_tech\_ai\_1\_df[region\_tech\_ai\_1\_df$ctry\_code == "AI\_pat",])

## # A tibble: 6 × 3  
## ctry\_code techn\_field\_nr n\_tech\_reg  
## <chr> <int> <dbl>  
## 1 AI\_pat 1 1.75  
## 2 AI\_pat 2 2.25  
## 3 AI\_pat 3 3.53  
## 4 AI\_pat 4 2.75  
## 5 AI\_pat 5 12.9   
## 6 AI\_pat 6 279.

We also save this file for later usage (“Files\_created\_with\_the\_code/data/files\_code\_Fields\_analysis/reg\_techAI\_FirstPeriod.csv”). We do the exact same thing for the two remaining intervals (namely Interval 2 [1989-2003], and Interval 3 [2004-2018]), saving corresponding interval-specific files throughout the process. At the end, we combine the three interval-specific files with countries’ general and AI-specific specialisations (namely rca\_data\_period\_1\_df, rca\_data\_period\_2\_df, rca\_data\_period\_3\_df) into a single file named ipc\_rcas\_df, which we also save (“Files\_created\_with\_the\_code/data/files\_code\_Fields\_analysis/IPC\_RCAs.csv”). Using again the case of Japan as an example and considering each of the three intervals, the combined and saved ipc\_rcas\_df file looks like this:

head(IPC\_RCAs[IPC\_RCAs$ctry\_code == "JP" & IPC\_RCAs$Period == "1974-1988",])

## ctry\_code techn\_field\_nr RCA\_Gen RCA\_AI Period  
## 2731 JP 1 1.1268685 1.4025974 1974-1988  
## 2732 JP 10 1.0109760 1.2022263 1974-1988  
## 2733 JP 11 0.7067914 0.0000000 1974-1988  
## 2734 JP 12 1.0644547 1.0957792 1974-1988  
## 2735 JP 13 0.6104061 0.7012987 1974-1988  
## 2736 JP 14 0.6177233 NA 1974-1988

head(IPC\_RCAs[IPC\_RCAs$ctry\_code == "JP" & IPC\_RCAs$Period == "1989-2003",])

## ctry\_code techn\_field\_nr RCA\_Gen RCA\_AI Period  
## 9486 JP 1 1.1395992 0.9834465 1989-2003  
## 9487 JP 10 1.0147327 0.8904603 1989-2003  
## 9488 JP 11 0.6029495 0.6173858 1989-2003  
## 9489 JP 12 1.0394907 0.9396071 1989-2003  
## 9490 JP 13 0.5833007 0.5805270 1989-2003  
## 9491 JP 14 0.6666774 1.8521575 1989-2003

head(IPC\_RCAs[IPC\_RCAs$ctry\_code == "JP" & IPC\_RCAs$Period == "2004-2018",])

## ctry\_code techn\_field\_nr RCA\_Gen RCA\_AI Period  
## 18341 JP 1 1.2953810 1.3244132 2004-2018  
## 18342 JP 10 0.8728225 0.9737969 2004-2018  
## 18343 JP 11 0.5286471 0.9540950 2004-2018  
## 18344 JP 12 0.8957888 1.1166269 2004-2018  
## 18345 JP 13 0.8071723 1.3124353 2004-2018  
## 18346 JP 14 0.5785004 3.3393324 2004-2018

Next, we create additional interval-specific files that combine the specialisations of the four considered countries with the specialisations of AI in a more user-friendly format. One file is created and saved per interval. For the first interval, the file created is named “First\_period”, and it is saved as “Files\_created\_with\_the\_code/data/files\_code\_Fields\_analysis/Metrics\_First\_period.csv”. The file looks like this:

head(First\_period)

## techn\_field\_nr sector field\_name  
## 1 1 Electrical engineering Electrical machinery, apparatus, energy  
## 2 2 Electrical engineering Audio-visual technology  
## 3 3 Electrical engineering Telecommunications  
## 4 4 Electrical engineering Digital communication  
## 5 5 Electrical engineering Basic communication processes  
## 6 6 Electrical engineering Computer technology  
## RCA\_US RCA\_CN RCA\_KR RCA\_JP RCA\_AI  
## 1 0.8061303 0.7812853 0.8859104 1.126869 0.05934926  
## 2 0.5290351 0.2812052 1.5817366 1.341886 0.08535377  
## 3 0.6577273 0.3468912 1.4004752 1.230989 0.33606440  
## 4 0.7356689 0.2069789 1.3043447 1.244439 1.09784935  
## 5 0.8740121 0.3793871 1.0785703 1.191508 1.64528254  
## 6 0.6008344 0.5349367 0.9466936 1.371771 16.64849577

The same is done for the remaining two intervals. Finally, we combine these 3 interval-specific files into a single file, adding some additional labels to the technological fields considering their visual location around the GTS. These additional labels are just information for analysis, which is not really used or mentioned in the paper. The file is named “All\_periods” and saved at “Files\_created\_with\_the\_code/data/files\_code\_Fields\_analysis/Specializations\_All\_periods\_IPC.csv”. It looks like this:

head(IPC\_names)

## techn\_field\_nr sector field\_name  
## 1 1 Electrical engineering Electrical machinery, apparatus, energy  
## 2 2 Electrical engineering Audio-visual technology  
## 3 3 Electrical engineering Telecommunications  
## 4 4 Electrical engineering Digital communication  
## 5 5 Electrical engineering Basic communication processes  
## 6 6 Electrical engineering Computer technology  
## RCA\_US RCA\_CN RCA\_KR RCA\_JP  
## 1 0,806130295464133 0,781285306548244 0,885910356664018 1,1268685164478  
## 2 0,52903509458199 0,281205160490934 1,58173657700963 1,34188556343943  
## 3 0,657727334104245 0,346891200617381 1,40047520647904 1,23098888970641  
## 4 0,735668946649889 0,206978925416903 1,30434472270759 1,2444394102031  
## 5 0,874012134211958 0,379387109049058 1,07857025021714 1,19150764942163  
## 6 0,600834418175826 0,534936673226541 0,946693574054904 1,37177058622865  
## RCA\_AI Period Category Category2  
## 1 0,0593492634039308 Period 1 (1974-1988) Surrounding fields 3  
## 2 0,0853537728458771 Period 1 (1974-1988) Surrounding fields 3  
## 3 0,336064398912961 Period 1 (1974-1988) Surrounding fields 3  
## 4 1,09784934911933 Period 1 (1974-1988) AI-related fields 2  
## 5 1,64528253859531 Period 1 (1974-1988) AI-related fields 2  
## 6 16,6484957668701 Period 1 (1974-1988) AI-core fields 1

In the last step, we use the previously saved file named “Files\_created\_with\_the\_code/data/files\_code\_Fields\_analysis/IPC\_RCAs.csv” to create a file summarizing the number of general (column Round\_general), AI-specific (column Round\_AI), and coinciding specialisations (column Total\_RCA) of each country over each interval. The previously calculated general and AI-specific specialisations are made binary, and their sum composes the number of coinciding specialisations. The resulting file IPC\_RCAs\_Top4 is saved at “Files\_created\_with\_the\_code/data/files\_code\_Fields\_analysis/RCA\_4countries\_detailed.csv” and looks like this:

head(IPC\_RCAs\_Top4)

## ctry\_code techn\_field\_nr RCA\_Gen RCA\_AI Period  
## 1 CN 1 0.7812853 0 1974-1988  
## 2 CN 10 1.2306427 0 1974-1988  
## 3 CN 11 0.9483087 0 1974-1988  
## 4 CN 12 0.7424070 0 1974-1988  
## 5 CN 13 1.3542371 0 1974-1988  
## 6 CN 14 0.9327427 0 1974-1988  
## Label Round\_general Round\_AI Total\_RCA  
## 1 Electrical machinery, apparatus, energy 0 0 0  
## 2 Measurement 1 0 1  
## 3 Analysis of biological materials 0 0 0  
## 4 Control 0 0 0  
## 5 Medical technology 1 0 1  
## 6 Organic fine chemistry 0 0 0

## 1.2. Create Sparse Matrix of relatedness between technological fields

This section creates a large sparse matrix from patent-techn\_field data and computes their co-occurrence (cross-product). Finally, it saves the resulting technology-relatedness matrix. We start again by reading the very large files containing patent and inventors’ location data. But this time, we apply the function “create\_sparse\_matrix”, which creates a sparse matrix of co-occurences of technological fields in patents. The result of applying the function is a matrix that shows in lines each unique application id of patents, and in columns all of the 35 possible technological fields containing information about the respective technological field being used in the respective patent or not (0s for not, above this value for the number of times that the code appears in each individual patent). Because the files are too big and a bit problematic computational-wise, they are split into smaller files that are summed up after everything is calculated. The first file resulting from applying the create\_sparse\_matrix function, named mat\_tech\_AI1, looks like this:

print(as.matrix(mat\_tech\_AI1[1:20, 1:12]))

## 1 2 3 4 5 6 7 8 9 10 11 12  
## 58 0 0 0 0 0 2 0 0 0 0 0 0  
## 76 0 0 0 0 0 0 0 0 0 0 0 0  
## 111 0 0 0 0 0 0 0 0 0 0 0 0  
## 139 0 0 0 0 0 0 0 0 0 0 0 4  
## 151 0 0 0 0 0 0 0 0 0 0 0 0  
## 159 0 0 0 0 0 0 0 0 0 0 0 0  
## 183 0 0 0 0 0 0 0 0 0 0 0 0  
## 193 0 0 0 0 0 0 0 0 0 0 0 0  
## 200 0 0 0 0 0 0 0 0 0 0 0 0  
## 206 0 0 0 0 0 0 0 0 0 0 0 0  
## 217 0 0 0 0 0 0 0 0 0 0 0 0  
## 218 0 0 0 0 0 0 0 0 0 0 0 0  
## 220 0 0 0 0 1 0 0 0 0 0 0 0  
## 231 1 0 0 0 0 0 0 0 0 0 0 0  
## 243 3 0 0 0 0 0 0 0 0 0 0 0  
## 246 0 0 0 0 0 0 0 0 0 0 0 0  
## 261 0 0 0 0 0 0 0 0 0 0 0 0  
## 266 0 0 0 0 0 0 0 0 0 0 0 0  
## 280 0 0 0 0 0 0 0 0 0 0 0 0  
## 283 0 0 0 0 0 0 0 0 0 0 0 0

This file is transformed in a square matrix of co-occurrences, which captures all possible combinations between two distinct technological fields. This square matrix looks like this:

print(as.matrix(mat\_tech\_AI1[1:35, 1:35]))

## 1 2 3 4 5 6 7 8 9  
## 1 5768289 180008 72869 35505 31634 105672 48931 293699 162448  
## 2 180008 2531644 182176 197948 36765 435699 37718 187833 317265  
## 3 72869 182176 2229334 679000 58436 355831 48692 13799 109871  
## 4 35505 197948 679000 3345610 45697 596334 160911 2506 4448  
## 5 31634 36765 58436 45697 506310 70807 435 34596 4931  
## 6 105672 435699 355831 596334 70807 6260090 553534 95310 109728  
## 7 48931 37718 48692 160911 435 553534 1701101 926 3984  
## 8 293699 187833 13799 2506 34596 95310 926 2907011 220961  
## 9 162448 317265 109871 4448 4931 109728 3984 220961 2596916  
## 10 224446 91013 138039 81955 26343 269221 40824 114839 106349  
## 11 6043 2012 4223 2157 380 26784 3310 2712 2321  
## 12 98397 90671 95899 110688 12005 283610 142899 10402 22262  
## 13 35221 33259 24034 12585 2645 127652 27438 12349 45652  
## 14 26284 5530 622 912 672 6859 840 102013 25607  
## 15 5711 1220 435 776 189 21330 1781 2263 3488  
## 16 2489 688 139 93 39 4725 836 495 1208  
## 17 110241 28068 1132 117 25 2902 20 78495 103336  
## 18 2856 155 61 77 9 1485 1255 128 28  
## 19 77403 28650 1243 108 222 5442 640 184747 86394  
## 20 250123 18549 2934 95 1357 10175 518 59457 25865  
## 21 124491 67392 4601 731 1170 14978 466 199001 88924  
## 22 58683 7150 2666 166 2069 3842 50 53804 25285  
## 23 82108 10257 2396 688 1471 10562 2764 45908 18939  
## 24 29344 9675 1464 973 442 7448 2572 14057 5130  
## 25 44158 19330 12865 4228 393 33521 15140 22677 47088  
## 26 57741 25924 2658 596 448 12340 988 62331 21196  
## 27 95078 5935 1948 1913 914 19023 2243 10748 8940  
## 28 21109 26245 76191 1919 819 80353 4467 16159 82423  
## 29 81035 28330 7767 3500 399 24244 6398 34466 75161  
## 30 73388 22945 6641 3584 535 12310 2500 19714 5256  
## 31 75597 20173 7014 913 583 15613 1003 5928 24813  
## 32 213765 54606 28756 21166 2988 70014 15738 6698 19059  
## 33 29184 20783 6655 8249 754 52055 14459 1232 4992  
## 34 26222 33300 13681 4925 1838 38240 5927 8552 13602  
## 35 59692 14997 14208 7775 743 35601 9893 7164 5614  
## 10 11 12 13 14 15 16 17 18  
## 1 224446 6043 98397 35221 26284 5711 2489 110241 2856  
## 2 91013 2012 90671 33259 5530 1220 688 28068 155  
## 3 138039 4223 95899 24034 622 435 139 1132 61  
## 4 81955 2157 110688 12585 912 776 93 117 77  
## 5 26343 380 12005 2645 672 189 39 25 9  
## 6 269221 26784 283610 127652 6859 21330 4725 2902 1485  
## 7 40824 3310 142899 27438 840 1781 836 20 1255  
## 8 114839 2712 10402 12349 102013 2263 495 78495 128  
## 9 106349 2321 22262 45652 25607 3488 1208 103336 28  
## 10 5893276 237979 213472 103998 43521 104810 29536 16762 5190  
## 11 237979 810267 8274 27172 48107 261182 96945 5074 7045  
## 12 213472 8274 2064137 43797 1902 1012 7803 1243 2340  
## 13 103998 27172 43797 2429739 16945 35668 125523 31251 7933  
## 14 43521 48107 1902 16945 2867623 218485 880680 142222 72141  
## 15 104810 261182 1012 35668 218485 2548809 536552 39796 301160  
## 16 29536 96945 7803 125523 880680 536552 3083488 65679 234071  
## 17 16762 5074 1243 31251 142222 39796 65679 2049722 20411  
## 18 5190 7045 2340 7933 72141 301160 234071 20411 1898838  
## 19 47689 10954 5778 25452 366020 132050 85334 343204 59224  
## 20 31626 5843 5974 19786 46278 8297 13210 68382 3902  
## 21 33031 4142 3131 31515 19938 5668 6219 104006 1959  
## 22 50026 9304 680 8844 9300 10986 17217 14426 774  
## 23 95527 24371 13263 59686 345426 43438 24794 94806 26817  
## 24 35712 6285 16084 46015 30433 67992 3463 28112 9471  
## 25 45455 2166 44920 48773 2091 2818 2974 20540 15336  
## 26 40922 1909 20787 14450 4094 10729 1429 9678 2731  
## 27 53426 2287 17820 20017 1825 2211 1446 4330 264  
## 28 18286 2120 8993 22087 14666 11092 4794 88256 2880  
## 29 43759 15562 26462 52776 19353 67652 34654 501786 128187  
## 30 27833 1142 23917 21157 2021 1160 355 2040 4135  
## 31 56718 1111 18569 14402 1537 1394 181 23407 2318  
## 32 138519 2364 132219 19788 1447 659 425 36789 268  
## 33 17664 730 46615 54425 1383 404 2645 6147 11201  
## 34 22110 1079 26491 44840 10727 5087 4469 22655 7095  
## 35 80910 8314 49748 10065 1578 1955 240 24444 750  
## 19 20 21 22 23 24 25 26 27  
## 1 77403 250123 124491 58683 82108 29344 44158 57741 95078  
## 2 28650 18549 67392 7150 10257 9675 19330 25924 5935  
## 3 1243 2934 4601 2666 2396 1464 12865 2658 1948  
## 4 108 95 731 166 688 973 4228 596 1913  
## 5 222 1357 1170 2069 1471 442 393 448 914  
## 6 5442 10175 14978 3842 10562 7448 33521 12340 19023  
## 7 640 518 466 50 2764 2572 15140 988 2243  
## 8 184747 59457 199001 53804 45908 14057 22677 62331 10748  
## 9 86394 25865 88924 25285 18939 5130 47088 21196 8940  
## 10 47689 31626 33031 50026 95527 35712 45455 40922 53426  
## 11 10954 5843 4142 9304 24371 6285 2166 1909 2287  
## 12 5778 5974 3131 680 13263 16084 44920 20787 17820  
## 13 25452 19786 31515 8844 59686 46015 48773 14450 20017  
## 14 366020 46278 19938 9300 345426 30433 2091 4094 1825  
## 15 132050 8297 5668 10986 43438 67992 2818 10729 2211  
## 16 85334 13210 6219 17217 24794 3463 2974 1429 1446  
## 17 343204 68382 104006 14426 94806 28112 20540 9678 4330  
## 18 59224 3902 1959 774 26817 9471 15336 2731 264  
## 19 3281112 126796 133673 28416 223167 106935 15567 32978 13952  
## 20 126796 3478542 211364 137164 225615 137648 8314 183631 36572  
## 21 133673 211364 1789373 35420 95519 27346 49898 77804 25745  
## 22 28416 137164 35420 405306 39145 5423 1106 4918 2601  
## 23 223167 225615 95519 39145 2731402 479200 47416 48496 45189  
## 24 106935 137648 27346 5423 479200 2069291 14231 24015 83658  
## 25 15567 8314 49898 1106 47416 14231 1786154 54803 10446  
## 26 32978 183631 77804 4918 48496 24015 54803 2329323 32652  
## 27 13952 36572 25745 2601 45189 83658 10446 32652 1816529  
## 28 101627 20432 67269 10598 78191 8483 51364 15113 2404  
## 29 217172 114726 135262 11263 90207 46258 57020 61081 17697  
## 30 28914 66431 13007 1510 51070 84902 9815 22827 74745  
## 31 20605 32638 32746 2071 31742 14969 60074 74156 123929  
## 32 9411 11411 24564 616 18160 28141 63146 34357 100390  
## 33 7559 2704 9689 149 23655 7024 39448 9904 5100  
## 34 37134 9178 43825 1395 36675 8805 41268 14452 7191  
## 35 77863 81833 44971 368 39360 52309 47536 31784 39369  
## 28 29 30 31 32 33 34 35  
## 1 21109 81035 73388 75597 213765 29184 26222 59692  
## 2 26245 28330 22945 20173 54606 20783 33300 14997  
## 3 76191 7767 6641 7014 28756 6655 13681 14208  
## 4 1919 3500 3584 913 21166 8249 4925 7775  
## 5 819 399 535 583 2988 754 1838 743  
## 6 80353 24244 12310 15613 70014 52055 38240 35601  
## 7 4467 6398 2500 1003 15738 14459 5927 9893  
## 8 16159 34466 19714 5928 6698 1232 8552 7164  
## 9 82423 75161 5256 24813 19059 4992 13602 5614  
## 10 18286 43759 27833 56718 138519 17664 22110 80910  
## 11 2120 15562 1142 1111 2364 730 1079 8314  
## 12 8993 26462 23917 18569 132219 46615 26491 49748  
## 13 22087 52776 21157 14402 19788 54425 44840 10065  
## 14 14666 19353 2021 1537 1447 1383 10727 1578  
## 15 11092 67652 1160 1394 659 404 5087 1955  
## 16 4794 34654 355 181 425 2645 4469 240  
## 17 88256 501786 2040 23407 36789 6147 22655 24444  
## 18 2880 128187 4135 2318 268 11201 7095 750  
## 19 101627 217172 28914 20605 9411 7559 37134 77863  
## 20 20432 114726 66431 32638 11411 2704 9178 81833  
## 21 67269 135262 13007 32746 24564 9689 43825 44971  
## 22 10598 11263 1510 2071 616 149 1395 368  
## 23 78191 90207 51070 31742 18160 23655 36675 39360  
## 24 8483 46258 84902 14969 28141 7024 8805 52309  
## 25 51364 57020 9815 60074 63146 39448 41268 47536  
## 26 15113 61081 22827 74156 34357 9904 14452 31784  
## 27 2404 17697 74745 123929 100390 5100 7191 39369  
## 28 1308730 42673 1881 8205 6575 13946 59257 7957  
## 29 42673 3126645 15300 49310 69364 18062 29144 90968  
## 30 1881 15300 1463621 30728 33966 23370 41410 31058  
## 31 8205 49310 30728 1831696 212337 19557 16086 103204  
## 32 6575 69364 33966 212337 2772775 40099 19890 113411  
## 33 13946 18062 23370 19557 40099 1236189 41595 36894  
## 34 59257 29144 41410 16086 19890 41595 1084525 19702  
## 35 7957 90968 31058 103204 113411 36894 19702 3560637

Six small co-occurrence matrices are calculated, and then they are summed up in a file named mat\_tech\_AI\_Final, which is saved at “Files\_created\_with\_the\_code/data/files\_code\_Fields\_analysis/Matrix\_IPC.csv”. This final file of co-occurrences between technological fields looks like this:

print(mat\_tech\_AI\_Final[1:35, 1:35])

## 1 10 11 12 13 14 15 16  
## 1 26242833 1105050 36467 460300 170587 113955 20807 10445  
## 10 1105050 24682955 1084761 1050741 563614 204173 409432 106582  
## 11 36467 1084761 3525353 34621 131313 253957 1242197 563632  
## 12 460300 1050741 34621 8999453 164516 8290 6446 22004  
## 13 170587 563614 131313 164516 10467208 97468 155323 563556  
## 14 113955 204173 253957 8290 97468 15116843 1152360 4615828  
## 15 20807 409432 1242197 6446 155323 1152360 10401074 2707298  
## 16 10445 106582 563632 22004 563556 4615828 2707298 13025114  
## 17 533998 77728 30836 7817 179377 854888 181385 316366  
## 18 13709 22386 30137 10141 44737 337538 1089819 896097  
## 19 443392 201039 52245 23760 152775 2235446 541154 390863  
## 2 1119478 563726 8619 438005 198152 47780 6197 4455  
## 20 1469546 174826 26056 38608 100743 231642 35052 67891  
## 21 724137 157456 18887 22339 153392 104417 22737 24447  
## 22 219144 173075 27249 3116 27220 31400 33307 56329  
## 23 419139 524854 149002 76178 325463 1755442 221998 152120  
## 24 145875 184676 37931 72363 213020 130941 260273 17747  
## 25 237008 244039 12659 304054 217590 15050 11041 15281  
## 26 327159 229919 7065 176239 74190 32237 36376 8564  
## 27 460192 330778 16227 119692 94111 13592 11044 7183  
## 28 142474 114862 13331 64315 127019 121893 50116 22389  
## 29 432784 238130 91847 145060 252392 105799 273218 160764  
## 3 362362 613474 16588 454421 86787 4913 2593 2314  
## 30 386632 135633 7282 131213 92050 13022 6068 2210  
## 31 400546 274385 7332 126129 71485 8727 5164 3323  
## 32 915437 662630 10983 612923 88497 7206 3185 3009  
## 33 146905 82825 3331 219052 261890 9877 1754 11232  
## 34 128408 127036 6630 145379 209610 61408 20616 18056  
## 35 278714 366014 36342 230550 43541 8633 10697 1797  
## 4 143077 325019 8870 479100 46635 4488 3065 1814  
## 5 202650 169190 1162 71090 15821 2653 992 290  
## 6 445672 1223744 111402 1314660 492137 32031 93726 20856  
## 7 189070 160542 16033 735780 120184 3834 7143 3661  
## 8 1432517 686541 12597 73831 50575 365327 9801 3727  
## 9 832079 615023 15757 123855 349238 252442 17505 11026  
## 17 18 19 2 20 21 22 23 24  
## 1 533998 13709 443392 1119478 1469546 724137 219144 419139 145875  
## 10 77728 22386 201039 563726 174826 157456 173075 524854 184676  
## 11 30836 30137 52245 8619 26056 18887 27249 149002 37931  
## 12 7817 10141 23760 438005 38608 22339 3116 76178 72363  
## 13 179377 44737 152775 198152 100743 153392 27220 325463 213020  
## 14 854888 337538 2235446 47780 231642 104417 31400 1755442 130941  
## 15 181385 1089819 541154 6197 35052 22737 33307 221998 260273  
## 16 316366 896097 390863 4455 67891 24447 56329 152120 17747  
## 17 10676610 93923 1944663 240044 362392 579773 46352 527703 125750  
## 18 93923 7317728 265045 1573 21057 15419 2554 153172 47742  
## 19 1944663 265045 15422287 356341 685512 733594 101637 1108478 494356  
## 2 240044 1573 356341 14335959 199276 506317 30510 68765 45202  
## 20 362392 21057 685512 199276 16386118 1158401 438789 1165654 675668  
## 21 579773 15419 733594 506317 1158401 9278629 112480 555537 145430  
## 22 46352 2554 101637 30510 438789 112480 1324026 133465 21055  
## 23 527703 153172 1108478 68765 1165654 555537 133465 13214230 2323343  
## 24 125750 47742 494356 45202 675668 145430 21055 2323343 9199630  
## 25 125085 85630 87337 147285 57671 295880 5819 276808 75744  
## 26 56172 14007 184865 248275 922532 425365 16226 263347 138952  
## 27 24044 2876 88232 31077 232388 135681 12957 265500 460881  
## 28 488151 16674 601362 296926 145835 373035 36658 430127 58240  
## 29 2274295 531516 1012771 232699 662971 890035 36419 548439 222304  
## 3 6304 725 9616 993581 24974 21894 10691 15323 13785  
## 30 11193 19936 146983 90727 463218 90091 4051 302186 446346  
## 31 125667 5232 126105 115040 200267 209204 9998 168414 85347  
## 32 157364 2287 59265 239762 71269 139276 3684 92185 133339  
## 33 37219 52233 42519 99615 19752 62699 315 126170 49967  
## 34 102186 39290 179676 210021 51377 239564 4544 174306 59730  
## 35 119196 2513 364466 88928 386575 262458 1693 215646 271865  
## 4 1398 458 1428 798019 895 1899 584 4346 5944  
## 5 869 190 1584 300865 7543 8101 6949 8945 5790  
## 6 11786 7036 31035 2073525 39318 55552 12485 52088 35730  
## 7 1054 4409 2739 166538 5415 1756 156 10773 19378  
## 8 379736 1018 703190 1132716 384304 1171768 184721 282961 73581  
## 9 640378 1250 591978 1778205 204091 496639 90594 135042 39531  
## 25 26 27 28 29 3 30 31 32  
## 1 237008 327159 460192 142474 432784 362362 386632 400546 915437  
## 10 244039 229919 330778 114862 238130 613474 135633 274385 662630  
## 11 12659 7065 16227 13331 91847 16588 7282 7332 10983  
## 12 304054 176239 119692 64315 145060 454421 131213 126129 612923  
## 13 217590 74190 94111 127019 252392 86787 92050 71485 88497  
## 14 15050 32237 13592 121893 105799 4913 13022 8727 7206  
## 15 11041 36376 11044 50116 273218 2593 6068 5164 3185  
## 16 15281 8564 7183 22389 160764 2314 2210 3323 3009  
## 17 125085 56172 24044 488151 2274295 6304 11193 125667 157364  
## 18 85630 14007 2876 16674 531516 725 19936 5232 2287  
## 19 87337 184865 88232 601362 1012771 9616 146983 126105 59265  
## 2 147285 248275 31077 296926 232699 993581 90727 115040 239762  
## 20 57671 922532 232388 145835 662971 24974 463218 200267 71269  
## 21 295880 425365 135681 373035 890035 21894 90091 209204 139276  
## 22 5819 16226 12957 36658 36419 10691 4051 9998 3684  
## 23 276808 263347 265500 430127 548439 15323 302186 168414 92185  
## 24 75744 138952 460881 58240 222304 13785 446346 85347 133339  
## 25 9267812 360892 62498 397629 374961 93698 55575 293059 316170  
## 26 360892 11407587 174310 109204 347920 16900 127266 387707 170519  
## 27 62498 174310 9032042 15843 88766 13427 374929 659505 542172  
## 28 397629 109204 15843 8187901 299707 477030 14857 59083 41079  
## 29 374961 347920 88766 299707 15132833 39136 87262 321723 405485  
## 3 93698 16900 13427 477030 39136 10554590 37350 34597 153809  
## 30 55575 127266 374929 14857 87262 37350 7500577 170350 167141  
## 31 293059 387707 659505 59083 321723 34597 170350 8975847 1094346  
## 32 316170 170519 542172 41079 405485 153809 167141 1094346 12649692  
## 33 190775 52267 27189 76670 114989 31671 134233 95790 175913  
## 34 221657 84352 35329 340332 184052 79752 209624 86359 120849  
## 35 262033 187657 178042 45916 470602 71338 166601 577745 544994  
## 4 15775 2672 6855 15540 13139 2820312 13797 3925 93460  
## 5 2484 4826 7045 6822 3479 406133 4839 5318 18938  
## 6 174082 64361 81588 501207 104021 1706302 53130 73536 275421  
## 7 109065 21471 11568 28035 33017 220822 12660 3825 64206  
## 8 139063 348827 64121 93685 168451 112782 101086 34116 32146  
## 9 447847 140327 60694 702628 462908 760568 24403 156239 92514  
## 33 34 35 4 5 6 7 8 9  
## 1 146905 128408 278714 143077 202650 445672 189070 1432517 832079  
## 10 82825 127036 366014 325019 169190 1223744 160542 686541 615023  
## 11 3331 6630 36342 8870 1162 111402 16033 12597 15757  
## 12 219052 145379 230550 479100 71090 1314660 735780 73831 123855  
## 13 261890 209610 43541 46635 15821 492137 120184 50575 349238  
## 14 9877 61408 8633 4488 2653 32031 3834 365327 252442  
## 15 1754 20616 10697 3065 992 93726 7143 9801 17505  
## 16 11232 18056 1797 1814 290 20856 3661 3727 11026  
## 17 37219 102186 119196 1398 869 11786 1054 379736 640378  
## 18 52233 39290 2513 458 190 7036 4409 1018 1250  
## 19 42519 179676 364466 1428 1584 31035 2739 703190 591978  
## 2 99615 210021 88928 798019 300865 2073525 166538 1132716 1778205  
## 20 19752 51377 386575 895 7543 39318 5415 384304 204091  
## 21 62699 239564 262458 1899 8101 55552 1756 1171768 496639  
## 22 315 4544 1693 584 6949 12485 156 184721 90594  
## 23 126170 174306 215646 4346 8945 52088 10773 282961 135042  
## 24 49967 59730 271865 5944 5790 35730 19378 73581 39531  
## 25 190775 221657 262033 15775 2484 174082 109065 139063 447847  
## 26 52267 84352 187657 2672 4826 64361 21471 348827 140327  
## 27 27189 35329 178042 6855 7045 81588 11568 64121 60694  
## 28 76670 340332 45916 15540 6822 501207 28035 93685 702628  
## 29 114989 184052 470602 13139 3479 104021 33017 168451 462908  
## 3 31671 79752 71338 2820312 406133 1706302 220822 112782 760568  
## 30 134233 209624 166601 13797 4839 53130 12660 101086 24403  
## 31 95790 86359 577745 3925 5318 73536 3825 34116 156239  
## 32 175913 120849 544994 93460 18938 275421 64206 32146 92514  
## 33 5758204 219012 209748 38328 3621 208961 55401 5833 22884  
## 34 219012 5290424 120742 30035 22367 242998 39460 45044 97315  
## 35 209748 120742 15805378 29060 4422 155283 59391 40477 32646  
## 4 38328 30035 29060 12132499 261134 2358962 648691 11314 27124  
## 5 3621 22367 4422 261134 2871031 435879 3267 212314 34443  
## 6 208961 242998 155283 2358962 435879 25287889 2302961 544407 544095  
## 7 55401 39460 59391 648691 3267 2302961 6633202 7694 19252  
## 8 5833 45044 40477 11314 212314 544407 7694 13624917 1366521  
## 9 22884 97315 32646 27124 34443 544095 19252 1366521 15211979

This matrix is used for calculating the relatedness between all technological fields, which is at the core of the GTS. The matrix is normalised to prevent overestimating knowledge links associated with ubiquitously used technological fields through the cosine index. This normalised matrix is then used as an input for calculating relatedness. Both relatedness and the normalisation are made using the function “relatedness” provided by the EconGeo package. The resulting matrix of relatedness looks like this:

print(mat\_tech\_rel\_AI[1:35, 1:35])

## 1 10 11 12 13 14  
## 1 0.0000000000 0.079946743 0.0047211202 0.0428721050 0.018407944 0.0082334420  
## 10 0.0799467434 0.000000000 0.1470137330 0.1024491448 0.063667816 0.0154427693  
## 11 0.0047211202 0.147013733 0.0000000000 0.0060405771 0.026544349 0.0343726668  
## 12 0.0428721050 0.102449145 0.0060405771 0.0000000000 0.023925516 0.0008072283  
## 13 0.0184079442 0.063667816 0.0265443487 0.0239255157 0.000000000 0.0109958659  
## 14 0.0082334420 0.015442769 0.0343726668 0.0008072283 0.010995866 0.0000000000  
## 15 0.0018598341 0.038311165 0.2079985053 0.0007765132 0.021678032 0.1076864208  
## 16 0.0008350303 0.008919828 0.0844100996 0.0023707689 0.070347746 0.3857904013  
## 17 0.0429860301 0.006550041 0.0046499763 0.0008480503 0.022546223 0.0719457183  
## 18 0.0018409602 0.003146980 0.0075812991 0.0018353260 0.009380475 0.0473881020  
## 19 0.0311910778 0.014804770 0.0068848147 0.0022525932 0.016780858 0.1644048918  
## 2 0.0820081944 0.043230333 0.0011827789 0.0432428754 0.022665196 0.0036592818  
## 20 0.1179295984 0.014686709 0.0039169892 0.0041755217 0.012623327 0.0194341238  
## 21 0.0604546814 0.013760913 0.0029537734 0.0025134297 0.019995453 0.0091135679  
## 22 0.0418364232 0.034589010 0.0097449704 0.0008017080 0.008113952 0.0062670424  
## 23 0.0305215958 0.040009811 0.0203257348 0.0074760559 0.037005828 0.1336422171  
## 24 0.0143538334 0.019022882 0.0069917569 0.0095961584 0.032728549 0.0134700966  
## 25 0.0260260685 0.028053279 0.0026040538 0.0449976769 0.037308160 0.0017277865  
## 26 0.0358266152 0.026357280 0.0014493193 0.0260101428 0.012685618 0.0036907068  
## 27 0.0557420751 0.041943002 0.0036820298 0.0195390301 0.017799336 0.0017212163  
## 28 0.0148884424 0.012565184 0.0026096446 0.0090577375 0.020725340 0.0133168172  
## 29 0.0333604224 0.019215570 0.0132626502 0.0150696036 0.030377726 0.0085260904  
## 3 0.0305611387 0.054162895 0.0026207519 0.0516510419 0.011428807 0.0004331932  
## 30 0.0501726563 0.018425254 0.0017702107 0.0229477428 0.018651442 0.0017666683  
## 31 0.0429894310 0.030828223 0.0014741333 0.0182439158 0.011979634 0.0009792246  
## 32 0.0878084290 0.066536150 0.0019734845 0.0792332506 0.013254264 0.0007226201  
## 33 0.0227148349 0.013406434 0.0009648341 0.0456471892 0.063228264 0.0015966366  
## 34 0.0169696429 0.017574633 0.0016413422 0.0258926266 0.043252574 0.0084842527  
## 35 0.0303031181 0.041658637 0.0074018853 0.0337821644 0.007391733 0.0009812921  
## 4 0.0131213618 0.031203022 0.0015238341 0.0592146365 0.006677908 0.0004302986  
## 5 0.0355792970 0.031095982 0.0003821746 0.0168210491 0.004337146 0.0004869631  
## 6 0.0294008866 0.084511241 0.0137671187 0.1168832653 0.050693284 0.0022091418  
## 7 0.0220325975 0.019584414 0.0034999582 0.1155539683 0.021868000 0.0004670928  
## 8 0.1174737987 0.058936797 0.0019351443 0.0081596924 0.006475845 0.0313206684  
## 9 0.0653998810 0.050603736 0.0023200157 0.0131195745 0.042860104 0.0207434998  
## 15 16 17 18 19  
## 1 0.0018598341 8.350303e-04 0.0429860301 1.840960e-03 0.0311910778  
## 10 0.0383111645 8.919828e-03 0.0065500411 3.146980e-03 0.0148047702  
## 11 0.2079985053 8.441010e-02 0.0046499763 7.581299e-03 0.0068848147  
## 12 0.0007765132 2.370769e-03 0.0008480503 1.835326e-03 0.0022525932  
## 13 0.0216780318 7.034775e-02 0.0225462230 9.380475e-03 0.0167808577  
## 14 0.1076864208 3.857904e-01 0.0719457183 4.738810e-02 0.1644048918  
## 15 0.0000000000 2.799333e-01 0.0188848613 1.892857e-01 0.0492366077  
## 16 0.2799332981 0.000000e+00 0.0294598999 1.392026e-01 0.0318068634  
## 17 0.0188848613 2.945990e-02 0.0000000000 1.469123e-02 0.1593435782  
## 18 0.1892856606 1.392026e-01 0.0146912335 0.000000e+00 0.0362293665  
## 19 0.0492366077 3.180686e-02 0.1593435782 3.622937e-02 0.0000000000  
## 2 0.0005871484 3.775228e-04 0.0204823710 2.239077e-04 0.0265711277  
## 20 0.0036381212 6.302395e-03 0.0338739562 3.283483e-03 0.0559960254  
## 21 0.0024550881 2.360959e-03 0.0563786757 2.501290e-03 0.0623400971  
## 22 0.0082240398 1.243972e-02 0.0103072100 9.474242e-04 0.0197505551  
## 23 0.0209085033 1.281413e-02 0.0447596222 2.167340e-02 0.0821634652  
## 24 0.0331238548 2.020067e-03 0.0144126126 9.128223e-03 0.0495141523  
## 25 0.0015681178 1.941114e-03 0.0159992079 1.827133e-02 0.0097621694  
## 26 0.0051521271 1.084869e-03 0.0071649717 2.980510e-03 0.0206064941  
## 27 0.0017301955 1.006477e-03 0.0033923330 6.769110e-04 0.0108785922  
## 28 0.0067735255 2.706463e-03 0.0594176564 3.385728e-03 0.0639664334  
## 29 0.0272392016 1.433517e-02 0.2041995717 7.961140e-02 0.0794646743  
## 3 0.0002828487 2.257584e-04 0.0006192853 1.188129e-04 0.0008255129  
## 30 0.0010184497 3.317534e-04 0.0016918564 5.026966e-03 0.0194150678  
## 31 0.0007168367 4.125660e-04 0.0157100740 1.091128e-03 0.0137766654  
## 32 0.0003951317 3.338748e-04 0.0175816980 4.262580e-04 0.0057864012  
## 33 0.0003507739 2.009018e-03 0.0067032496 1.569338e-02 0.0066920404  
## 34 0.0035237878 2.760298e-03 0.0157296734 1.008930e-02 0.0241698078  
## 35 0.0015042335 2.260114e-04 0.0150951663 5.309084e-04 0.0403355572  
## 4 0.0003635501 1.924421e-04 0.0001493358 8.161560e-05 0.0001333030  
## 5 0.0002252616 5.889827e-05 0.0001777126 6.481909e-05 0.0002830795  
## 6 0.0079970513 1.591585e-03 0.0009056475 9.019231e-04 0.0020840078  
## 7 0.0010765853 4.935107e-04 0.0001430643 9.983476e-04 0.0003248910  
## 8 0.0010395284 3.535529e-04 0.0362719317 1.622137e-04 0.0586970696  
## 9 0.0017795051 1.002499e-03 0.0586268399 1.909066e-04 0.0473609570  
## 2 20 21 22 23 24  
## 1 0.0820081944 1.179296e-01 0.0604546814 4.183642e-02 0.030521596 0.0143538334  
## 10 0.0432303328 1.468671e-02 0.0137609127 3.458901e-02 0.040009811 0.0190228822  
## 11 0.0011827789 3.916989e-03 0.0029537734 9.744970e-03 0.020325735 0.0069917569  
## 12 0.0432428754 4.175522e-03 0.0025134297 8.017080e-04 0.007476056 0.0095961584  
## 13 0.0226651961 1.262333e-02 0.0199954528 8.113952e-03 0.037005828 0.0327285487  
## 14 0.0036592818 1.943412e-02 0.0091135679 6.267042e-03 0.133642217 0.0134700966  
## 15 0.0005871484 3.638121e-03 0.0024550881 8.224040e-03 0.020908503 0.0331238548  
## 16 0.0003775228 6.302395e-03 0.0023609594 1.243972e-02 0.012814133 0.0020200668  
## 17 0.0204823710 3.387396e-02 0.0563786757 1.030721e-02 0.044759622 0.0144126126  
## 18 0.0002239077 3.283483e-03 0.0025012897 9.474242e-04 0.021673405 0.0091282229  
## 19 0.0265711277 5.599603e-02 0.0623400971 1.975056e-02 0.082163465 0.0495141523  
## 2 0.0000000000 1.695104e-02 0.0448057108 6.174031e-03 0.005307846 0.0047146162  
## 20 0.0169510379 0.000000e+00 0.1122968912 9.727031e-02 0.098564026 0.0772004791  
## 21 0.0448057108 1.122969e-01 0.0000000000 2.593997e-02 0.048868761 0.0172866251  
## 22 0.0061740314 9.727031e-02 0.0259399686 0.000000e+00 0.026847354 0.0057230458  
## 23 0.0053078455 9.856403e-02 0.0488687610 2.684735e-02 0.000000000 0.2408848595  
## 24 0.0047146162 7.720048e-02 0.0172866251 5.723046e-03 0.240884859 0.0000000000  
## 25 0.0171437469 7.353645e-03 0.0392491583 1.765139e-03 0.032028260 0.0118424295  
## 26 0.0288191762 1.173081e-01 0.0562701192 4.908438e-03 0.030386762 0.0216650013  
## 27 0.0039901095 3.268567e-02 0.0198532805 4.335443e-03 0.033885799 0.0794840307  
## 28 0.0328899748 1.769600e-02 0.0470904028 1.058199e-02 0.047360851 0.0086652685  
## 29 0.0190132573 5.934095e-02 0.0828774930 7.754848e-03 0.044544882 0.0243980228  
## 3 0.0888243066 2.445764e-03 0.0022305973 2.490750e-03 0.001361695 0.0016553151  
## 30 0.0124797968 6.979983e-02 0.0141227698 1.452167e-03 0.041319283 0.0824684735  
## 31 0.0130875946 2.495849e-02 0.0271236703 2.964199e-03 0.019045686 0.0130420112  
## 32 0.0243775335 7.937944e-03 0.0161381299 9.761404e-04 0.009317023 0.0182100840  
## 33 0.0163267375 3.546367e-03 0.0117112377 1.345453e-04 0.020555983 0.0110002687  
## 34 0.0294201584 7.884043e-03 0.0382447082 1.658838e-03 0.024271806 0.0112388052  
## 35 0.0102486986 4.880473e-02 0.0344712891 5.084760e-04 0.024704671 0.0420851466  
## 4 0.0775753926 9.530852e-05 0.0002103794 1.479472e-04 0.000419960 0.0007761308  
## 5 0.0559917559 1.537781e-03 0.0017181401 3.370218e-03 0.001654780 0.0014473592  
## 6 0.1449959984 3.011869e-03 0.0044270452 2.275195e-03 0.003620695 0.0033560243  
## 7 0.0205711270 7.327252e-04 0.0002471933 5.021721e-05 0.001322784 0.0032151337  
## 8 0.0984609286 3.659449e-02 0.1160785667 4.184489e-02 0.024449892 0.0085912019  
## 9 0.1481480323 1.862670e-02 0.0471544184 1.966966e-02 0.011183826 0.0044238176  
## 25 26 27 28 29 3  
## 1 0.0260260685 0.0358266152 0.055742075 0.014888442 0.0333604224 0.0305611387  
## 10 0.0280532786 0.0263572803 0.041943002 0.012565184 0.0192155705 0.0541628953  
## 11 0.0026040538 0.0014493193 0.003682030 0.002609645 0.0132626502 0.0026207519  
## 12 0.0449976769 0.0260101428 0.019539030 0.009057738 0.0150696036 0.0516510419  
## 13 0.0373081598 0.0126856175 0.017799336 0.020725340 0.0303777262 0.0114288069  
## 14 0.0017277865 0.0036907068 0.001721216 0.013316817 0.0085260904 0.0004331932  
## 15 0.0015681178 0.0051521271 0.001730196 0.006773526 0.0272392016 0.0002828487  
## 16 0.0019411142 0.0010848691 0.001006477 0.002706463 0.0143351718 0.0002257584  
## 17 0.0159992079 0.0071649717 0.003392333 0.059417656 0.2041995717 0.0006192853  
## 18 0.0182713275 0.0029805103 0.000676911 0.003385728 0.0796114040 0.0001188129  
## 19 0.0097621694 0.0206064941 0.010878592 0.063966433 0.0794646743 0.0008255129  
## 2 0.0171437469 0.0288191762 0.003990110 0.032889975 0.0190132573 0.0888243066  
## 20 0.0073536449 0.1173080886 0.032685669 0.017695997 0.0593409491 0.0024457643  
## 21 0.0392491583 0.0562701192 0.019853281 0.047090403 0.0828774930 0.0022305973  
## 22 0.0017651388 0.0049084381 0.004335443 0.010581990 0.0077548482 0.0024907501  
## 23 0.0320282596 0.0303867619 0.033885799 0.047360851 0.0445448823 0.0013616949  
## 24 0.0118424295 0.0216650013 0.079484031 0.008665269 0.0243980228 0.0016553151  
## 25 0.0000000000 0.0627956705 0.012028618 0.066023302 0.0459252980 0.0125563307  
## 26 0.0627956705 0.0000000000 0.033455941 0.018082525 0.0424958601 0.0022585020  
## 27 0.0120286178 0.0334559414 0.000000000 0.002901717 0.0119925362 0.0019847688  
## 28 0.0660233018 0.0180825252 0.002901717 0.000000000 0.0349325957 0.0608339761  
## 29 0.0459252980 0.0424958601 0.011992536 0.034932596 0.0000000000 0.0036814901  
## 3 0.0125563307 0.0022585020 0.001984769 0.060833976 0.0036814901 0.0000000000  
## 30 0.0114592003 0.0261690874 0.085275127 0.002915238 0.0126303325 0.0059148913  
## 31 0.0499769675 0.0659355790 0.124059902 0.009588385 0.0385133884 0.0045314230  
## 32 0.0481874499 0.0259171598 0.091148363 0.005958010 0.0433813298 0.0180043047  
## 33 0.0468705456 0.0128058221 0.007368351 0.017925534 0.0198312289 0.0059761521  
## 34 0.0465444187 0.0176637492 0.008183065 0.068007535 0.0271294809 0.0128620166  
## 35 0.0452678703 0.0323295893 0.033927749 0.007548596 0.0570693104 0.0094653423  
## 4 0.0022987090 0.0003882866 0.001101845 0.002154933 0.0013439766 0.3156407767  
## 5 0.0006929596 0.0013425948 0.002167883 0.001811074 0.0006812798 0.0870173583  
## 6 0.0182475467 0.0067278259 0.009433556 0.049996066 0.0076539641 0.1373687157  
## 7 0.0201945425 0.0039646261 0.002362683 0.004939890 0.0042914199 0.0314030822  
## 8 0.0181200070 0.0453271283 0.009216066 0.011616757 0.0154076053 0.0112867234  
## 9 0.0559303588 0.0174767439 0.008361077 0.083504807 0.0405814354 0.0729520175  
## 30 31 32 33 34 35  
## 1 0.0501726563 0.0429894310 0.0878084290 0.0227148349 0.016969643 0.0303031181  
## 10 0.0184252544 0.0308282228 0.0665361497 0.0134064341 0.017574633 0.0416586371  
## 11 0.0017702107 0.0014741333 0.0019734845 0.0009648341 0.001641342 0.0074018853  
## 12 0.0229477428 0.0182439158 0.0792332506 0.0456471892 0.025892627 0.0337821644  
## 13 0.0186514425 0.0119796341 0.0132542643 0.0632282639 0.043252574 0.0073917329  
## 14 0.0017666683 0.0009792246 0.0007226201 0.0015966366 0.008484253 0.0009812921  
## 15 0.0010184497 0.0007168367 0.0003951317 0.0003507739 0.003523788 0.0015042335  
## 16 0.0003317534 0.0004125660 0.0003338748 0.0020090181 0.002760298 0.0002260114  
## 17 0.0016918564 0.0157100740 0.0175816980 0.0067032496 0.015729673 0.0150951663  
## 18 0.0050269657 0.0010911278 0.0004262580 0.0156933830 0.010089305 0.0005309084  
## 19 0.0194150678 0.0137766654 0.0057864012 0.0066920404 0.024169808 0.0403355572  
## 2 0.0124797968 0.0130875946 0.0243775335 0.0163267375 0.029420158 0.0102486986  
## 20 0.0697998256 0.0249584897 0.0079379438 0.0035463672 0.007884043 0.0488047296  
## 21 0.0141227698 0.0271236703 0.0161381299 0.0117112377 0.038244708 0.0344712891  
## 22 0.0014521667 0.0029641991 0.0009761404 0.0001345453 0.001658838 0.0005084760  
## 23 0.0413192832 0.0190456864 0.0093170225 0.0205559831 0.024271806 0.0247046714  
## 24 0.0824684735 0.0130420112 0.0182100840 0.0110002687 0.011238805 0.0420851466  
## 25 0.0114592003 0.0499769675 0.0481874499 0.0468705456 0.046544419 0.0452678703  
## 26 0.0261690874 0.0659355790 0.0259171598 0.0128058221 0.017663749 0.0323295893  
## 27 0.0852751274 0.1240599023 0.0911483634 0.0073683507 0.008183065 0.0339277487  
## 28 0.0029152382 0.0095883850 0.0059580101 0.0179255338 0.068007535 0.0075485964  
## 29 0.0126303325 0.0385133884 0.0433813298 0.0198312289 0.027129481 0.0570693104  
## 3 0.0059148913 0.0045314230 0.0180043047 0.0059761521 0.012862017 0.0094653423  
## 30 0.0000000000 0.0343305460 0.0301037097 0.0389727974 0.052017671 0.0340122540  
## 31 0.0343305460 0.0000000000 0.1630165414 0.0230018335 0.017723827 0.0975515119  
## 32 0.0301037097 0.1630165414 0.0000000000 0.0377518870 0.022166208 0.0822409234  
## 33 0.0389727974 0.0230018335 0.0377518870 0.0000000000 0.064756224 0.0510222196  
## 34 0.0520176706 0.0177238269 0.0221662080 0.0647562240 0.000000000 0.0251031082  
## 35 0.0340122540 0.0975515119 0.0822409234 0.0510222196 0.025103108 0.0000000000  
## 4 0.0023758719 0.0005590080 0.0118960447 0.0078642666 0.005267170 0.0041926944  
## 5 0.0015952753 0.0014500012 0.0046147969 0.0014223690 0.007509294 0.0012214002  
## 6 0.0065813376 0.0075338046 0.0252179541 0.0308420544 0.030654086 0.0161160090  
## 7 0.0027701677 0.0006922194 0.0103845197 0.0144442057 0.008793075 0.0108881031  
## 8 0.0155654441 0.0043447882 0.0036587766 0.0010702031 0.007063481 0.0052220143  
## 9 0.0036015122 0.0190708971 0.0100922402 0.0040241793 0.014626243 0.0040367410  
## 4 5 6 7 8  
## 1 1.312136e-02 3.557930e-02 0.0294008866 2.203260e-02 0.1174737987  
## 10 3.120302e-02 3.109598e-02 0.0845112408 1.958441e-02 0.0589367973  
## 11 1.523834e-03 3.821746e-04 0.0137671187 3.499958e-03 0.0019351443  
## 12 5.921464e-02 1.682105e-02 0.1168832653 1.155540e-01 0.0081596924  
## 13 6.677908e-03 4.337146e-03 0.0506932843 2.186800e-02 0.0064758451  
## 14 4.302986e-04 4.869631e-04 0.0022091418 4.670928e-04 0.0313206684  
## 15 3.635501e-04 2.252616e-04 0.0079970513 1.076585e-03 0.0010395284  
## 16 1.924421e-04 5.889827e-05 0.0015915850 4.935107e-04 0.0003535529  
## 17 1.493358e-04 1.777126e-04 0.0009056475 1.430643e-04 0.0362719317  
## 18 8.161560e-05 6.481909e-05 0.0009019231 9.983476e-04 0.0001622137  
## 19 1.333030e-04 2.830795e-04 0.0020840078 3.248910e-04 0.0586970696  
## 2 7.757539e-02 5.599176e-02 0.1449959984 2.057113e-02 0.0984609286  
## 20 9.530852e-05 1.537781e-03 0.0030118692 7.327252e-04 0.0365944939  
## 21 2.103794e-04 1.718140e-03 0.0044270452 2.471933e-04 0.1160785667  
## 22 1.479472e-04 3.370218e-03 0.0022751947 5.021721e-05 0.0418448908  
## 23 4.199600e-04 1.654780e-03 0.0036206946 1.322784e-03 0.0244498923  
## 24 7.761308e-04 1.447359e-03 0.0033560243 3.215134e-03 0.0085912019  
## 25 2.298709e-03 6.929596e-04 0.0182475467 2.019454e-02 0.0181200070  
## 26 3.882866e-04 1.342595e-03 0.0067278259 3.964626e-03 0.0453271283  
## 27 1.101845e-03 2.167883e-03 0.0094335555 2.362683e-03 0.0092160656  
## 28 2.154933e-03 1.811074e-03 0.0499960658 4.939890e-03 0.0116167573  
## 29 1.343977e-03 6.812798e-04 0.0076539641 4.291420e-03 0.0154076053  
## 3 3.156408e-01 8.701736e-02 0.1373687157 3.140308e-02 0.0112867234  
## 30 2.375872e-03 1.595275e-03 0.0065813376 2.770168e-03 0.0155654441  
## 31 5.590080e-04 1.450001e-03 0.0075338046 6.922194e-04 0.0043447882  
## 32 1.189604e-02 4.614797e-03 0.0252179541 1.038452e-02 0.0036587766  
## 33 7.864267e-03 1.422369e-03 0.0308420544 1.444421e-02 0.0010702031  
## 34 5.267170e-03 7.509294e-03 0.0306540862 8.793075e-03 0.0070634809  
## 35 4.192694e-03 1.221400e-03 0.0161160090 1.088810e-02 0.0052220143  
## 4 0.000000e+00 6.083916e-02 0.2065071466 1.003113e-01 0.0012311939  
## 5 6.083916e-02 0.000000e+00 0.0730502580 9.671703e-04 0.0442313905  
## 6 2.065071e-01 7.305026e-02 0.0000000000 2.561738e-01 0.0426157350  
## 7 1.003113e-01 9.671703e-04 0.2561738018 0.000000e+00 0.0010638894  
## 8 1.231194e-03 4.423139e-02 0.0426157350 1.063889e-03 0.0000000000  
## 9 2.829015e-03 6.877397e-03 0.0408218033 2.551475e-03 0.1274470866  
## 9  
## 1 0.0653998810  
## 10 0.0506037358  
## 11 0.0023200157  
## 12 0.0131195745  
## 13 0.0428601043  
## 14 0.0207434998  
## 15 0.0017795051  
## 16 0.0010024994  
## 17 0.0586268399  
## 18 0.0001909066  
## 19 0.0473609570  
## 2 0.1481480323  
## 20 0.0186266975  
## 21 0.0471544184  
## 22 0.0196696563  
## 23 0.0111838259  
## 24 0.0044238176  
## 25 0.0559303588  
## 26 0.0174767439  
## 27 0.0083610766  
## 28 0.0835048067  
## 29 0.0405814354  
## 3 0.0729520175  
## 30 0.0036015122  
## 31 0.0190708971  
## 32 0.0100922402  
## 33 0.0040241793  
## 34 0.0146262429  
## 35 0.0040367410  
## 4 0.0028290146  
## 5 0.0068773971  
## 6 0.0408218033  
## 7 0.0025514749  
## 8 0.1274470866  
## 9 0.0000000000

Next, the matrix is turned into a network (file g\_tech\_AI). The degree of centrality of nodes is calculated using the Eigenvector centrality of vertices (centrality\_eigen), which also calculates the width of the links between nodes (i.e., technological fields). Links that have below average width are excluded for better visualisation. Then, a network layout is calculated based on this network (coords\_tech\_AI). The resulting coordinates look like this for the top 10 technological fields:

coords\_tech\_AI[1:10,]

## x y  
## 1 135.6454 65.62816  
## 2 134.0933 64.22519  
## 3 135.2521 58.81728  
## 4 131.3472 67.12986  
## 5 135.1214 62.67884  
## 6 138.8334 59.25243  
## 7 137.6120 58.32931  
## 8 138.0070 57.48352  
## 9 136.7269 61.15738  
## 10 139.9433 57.74551

Then, the previously file with general, AI-specific and coinciding country specialisations (from the file “RCA\_4countries\_detailed.csv”) and the file with AI-specific specialisations (file “Specializations\_All\_periods\_IPC.csv”) are loaded. The resulting file is processed for facilitating the plotting later. A new category is created, reflecting the types of specialisations used in the paper (Var1, which goes from 0 to 3; 0 stands for no specialisation, 1 for general specialisation, 2 for AI-specific specialization, and 3 for coinciding specialization). This new dataset named Newtable is saved at “Files\_created\_with\_the\_code/data/files\_code\_Fields\_analysis/Table\_appendix.xlsx” and looks like this:

Newtable[1:10,]

## Var1 Var2 Var3 Freq  
## 1 No specialization CN 1974-1988 19  
## 2 General specialization CN 1974-1988 16  
## 3 AI-specific specialization CN 1974-1988 0  
## 4 Coinciding specialization CN 1974-1988 0  
## 5 No specialization JP 1974-1988 12  
## 6 General specialization JP 1974-1988 7  
## 7 AI-specific specialization JP 1974-1988 6  
## 8 Coinciding specialization JP 1974-1988 10  
## 9 No specialization KR 1974-1988 21  
## 10 General specialization KR 1974-1988 14

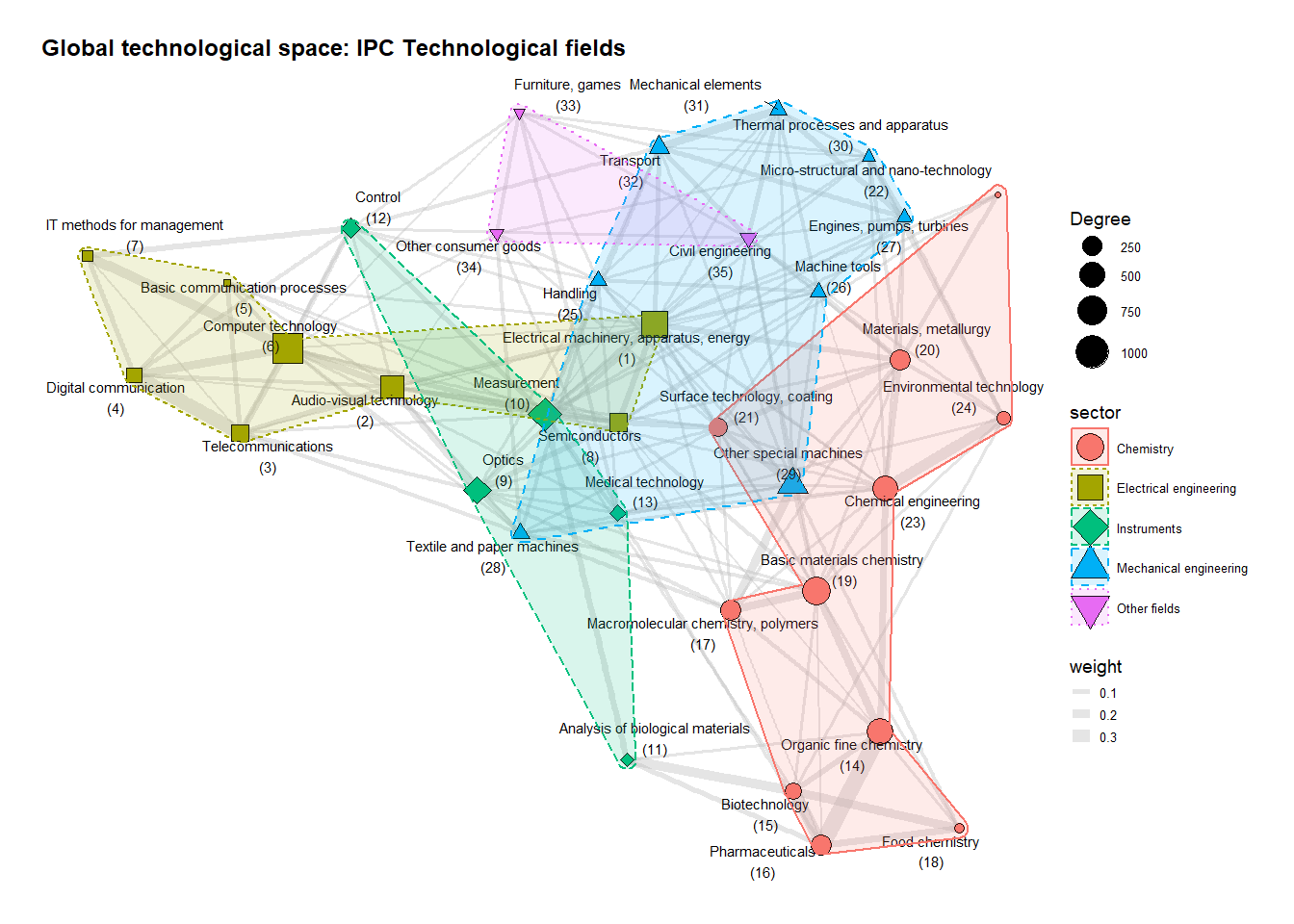
## 1.3. Plotting technological spaces

Next, we use the loaded information to plot technological spaces. For now, we have only calculated the coordinates and network for the GTS and not yet for the ATS, so we’ll start with this one.

### 1.3.1. Global technological space (GTS)

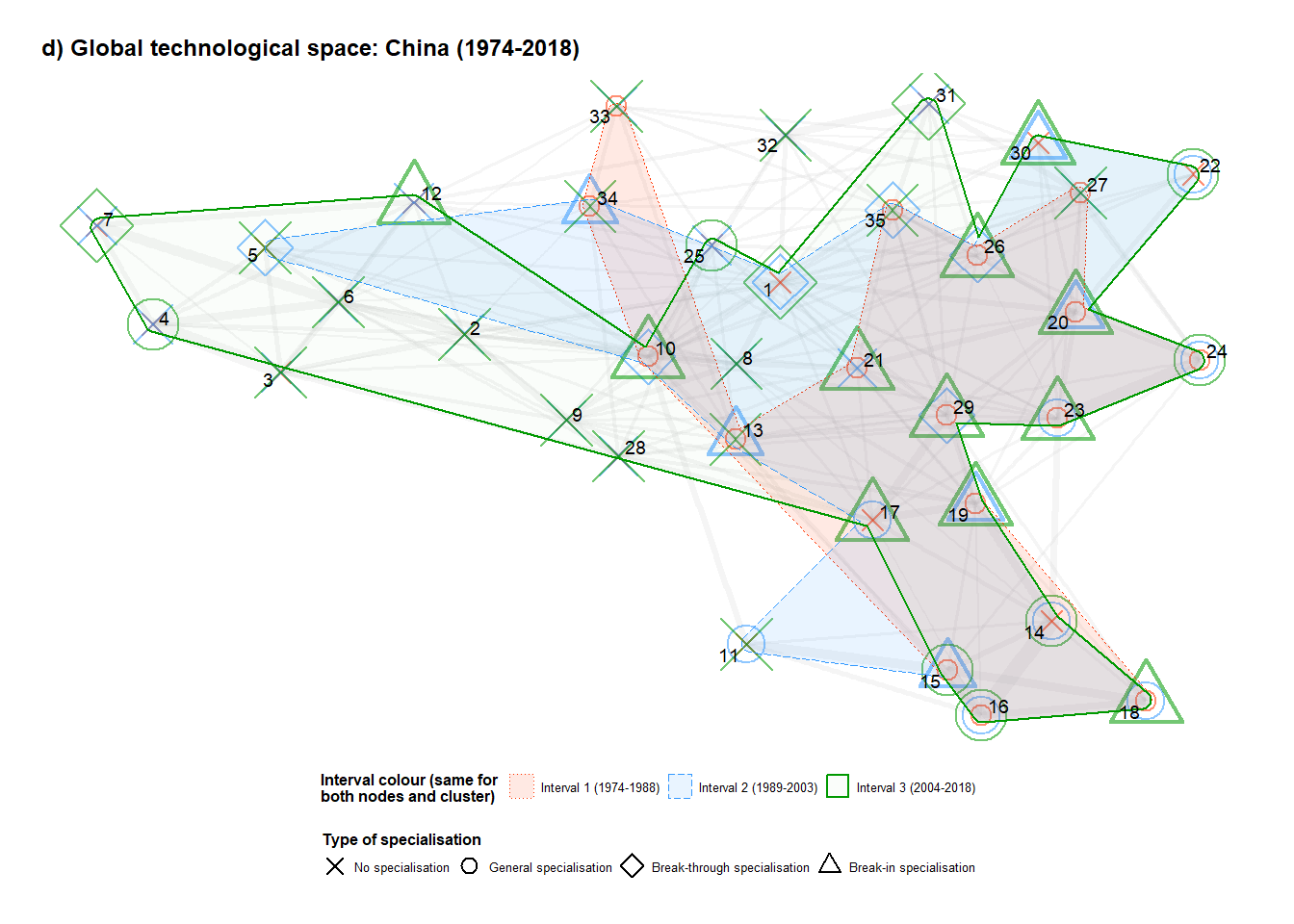
The first figure from the GTS doesn’t have any information linked to specialisations. The command below uses the general network created before to plot this geography-agnostic GTS. The figure produced is saved at “Files\_created\_with\_the\_code/figures/Figure\_3\_GTS\_for\_IPC\_fields.jpg”

g\_tech\_AI %>% ggraph(layout = coords\_tech\_AI) +   
 geom\_edge\_link(aes(width = weight), alpha = 0.4, colour = "grey") +   
 geom\_node\_point(aes(fill = sector, size = 1000^dgr, shape= sector))+ #   
 scale\_shape\_manual(values=c(21, 22, 23, 24, 25)) + scale\_size("Degree", range = c(2, 12)) +   
 geom\_node\_text(aes(label = paste0(field\_name, "\n(", name, ")")), size = 4, repel = TRUE) + #field\_name or name  
 theme\_graph(base\_family = "sans")+ ggtitle("Global technological space: IPC Technological fields") +   
 theme(legend.title = element\_text(size = 14), legend.text = element\_text(size = 10)) +   
 guides(colour = guide\_legend(override.aes = list(size=10)))+  
 geom\_mark\_hull(aes(x = x, y=y, colour = sector, fill= sector,  
 linetype = sector), alpha = 0.15, expand = unit(2.5, "mm"), size = 1)

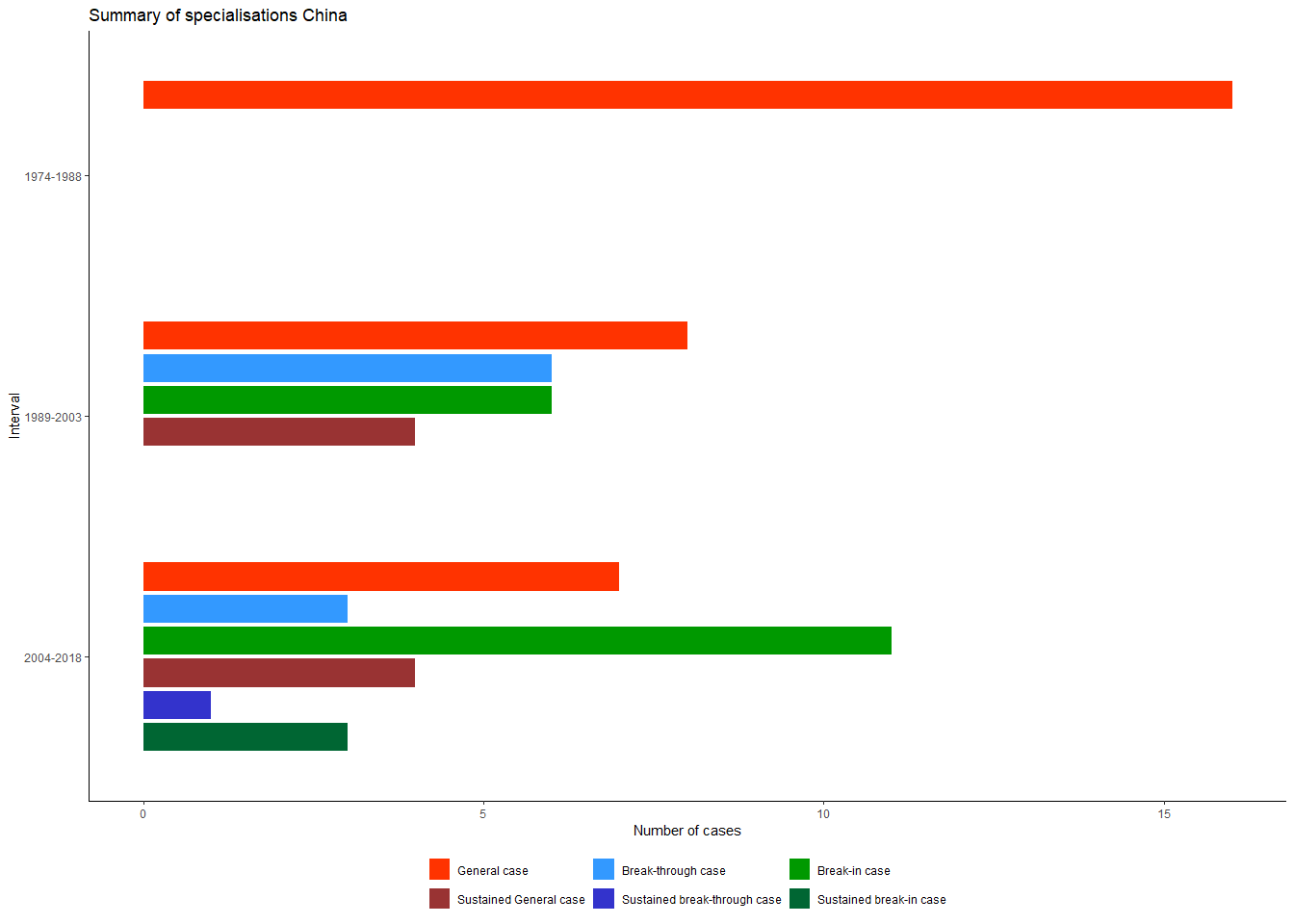


Next, this very same technological space is used to plot the technological trajectories of the selected countries, through plotting their specialisations information. Thus, the commands for color and size of the nodes are adapted to reflected the previously calculated 3 types of specialisations. Three technological spaces are plotted for each country (one for each interval). Using again the case of Japan as an example, in the first interval:

#GTS with specialisations per country  
country\_select <- c("CN", "US", "JP", "KR")  
### 1.2.3.3. Third Country  
i=1  
IPC\_RCAs\_wide\_simplified <- IPC\_RCAs\_Top4 %>% pivot\_wider(id\_cols = c(ctry\_code, techn\_field\_nr, Label),   
 names\_from = Period\_sim,  
 values\_from = c(RCA\_AI\_Period, Total\_RCA\_2, RCA\_Gen, RCA\_AI, Round\_general, Round\_AI, Total\_RCA),   
 names\_glue = "{.value}\_Period\_{Period\_sim}" )  
  
 g\_tech\_AI %N>% left\_join(IPC\_RCAs\_wide\_simplified %>%  
 filter(ctry\_code == country\_select[i]) %>%  
 select(-ctry\_code), by = c("name" = "techn\_field\_nr")) %>%  
 mutate(Shape\_Group\_P1\_Factor = factor(  
 ifelse(is.na(Total\_RCA\_2\_Period\_1), "NA\_Value", as.character(Total\_RCA\_2\_Period\_1)),  
 levels = c("0", "1", "2", "3", "NA\_Value"))) %>% ggraph(layout = coords\_tech\_AI) +  
 geom\_edge\_link(aes(width = weight), alpha = 0.2, colour = "#CCCCCC", show.legend = FALSE) +   
 geom\_node\_point(aes(shape = Shape\_Group\_P1\_Factor,   
 size = 5, stroke = ifelse(Total\_RCA\_2\_Period\_1 == 3, 2.5, 1.3),  
 alpha = 1), color = "#FF3300", show.legend = c(shape=TRUE, size=FALSE, stroke=FALSE, alpha=FALSE, color=FALSE)) +   
 geom\_node\_point(aes(shape = factor(Total\_RCA\_2\_Period\_2),  
 size = 5.5, stroke = ifelse(Total\_RCA\_2\_Period\_2 == 3, 2.5, 1.3),  
 alpha = 1), color = "#3399FF", show.legend = FALSE) +  
 geom\_node\_point(aes(shape = factor(Total\_RCA\_2\_Period\_3),   
 size = 6.5,stroke = ifelse(Total\_RCA\_2\_Period\_3 == 3, 2.5, 1.3),  
 alpha = 1), color = "#009900", show.legend = FALSE) +  
 scale\_shape\_manual(name = "Type of specialisation",  
 values = c("0" = 4, "1" = 1, "2" = 5, "3" = 2, "NA\_Value" = 16), breaks = c("0", "1", "2", "3"),   
 labels = c("0" = "No specialisation", "1" = "General specialisation",   
 "2" = "Break-through specialisation", "3" = "Break-in specialisation"),   
 na.translate = FALSE, drop = FALSE) + scale\_size("Degree", range = c(7, 18))+   
 scale\_alpha(guide = "none") +   
 #geom\_node\_label(aes(label = name), size = 2, repel = F) +   
 geom\_mark\_hull(aes(filter = Total\_RCA\_2\_Period\_1 > .99, x = x, y = y, fill = "Period 1", group = "Period 1"),   
 concavity = .1, alpha = .11, linetype = "dotted",expand = unit(2, "mm"), size = .5, color = "#FF3300") +   
 geom\_mark\_hull(aes(filter = Total\_RCA\_2\_Period\_2 > .99, x = x, y = y, fill = "Period 2", group = "Period 2"),  
 concavity = .1, alpha = .11, linetype = "longdash",expand = unit(2, "mm"), size = .5, color = "#3399FF") +  
 geom\_mark\_hull(aes(filter = Total\_RCA\_2\_Period\_3 > .99, x = x, y = y, fill = "Period 3", group = "Period 3"),  
 concavity = .1, alpha = .02, expand = unit(2, "mm"), size = 1, color = "#009900") +  
 scale\_fill\_manual(name = "Interval colour (same for \nboth nodes and cluster)", # New legend for fill  
 values = c("Period 1" = "#FF3300", "Period 2" = "#3399FF", "Period 3" = "#009900"),  
 labels = c("Interval 1 (1974-1988)", "Interval 2 (1989-2003)", "Interval 3 (2004-2018)")) +  
 theme\_graph(base\_family = "sans") + theme(legend.position = "bottom", #right  
 legend.box = "vertical", legend.title = element\_text(size = 12, face = "bold"),   
 legend.text = element\_text(size = 10), legend.key.size = unit(0.7, "cm") ) +  
 ggtitle("d) Global technological space: China (1974-2018)") +  
 geom\_node\_text(aes(label = name), size = 5, repel = TRUE) + #field\_name or name  
 guides(shape = guide\_legend(title.position = "top",   
 override.aes = list(size = 5, stroke = 1.5, color = "black") ),  
 colour = guide\_legend(title.position = "top",   
 override.aes = list(linetype = c("solid", "longdash", "dotted"),   
 alpha = 1, size = 1, shape = NA) ))



bar\_plot\_China <- bar\_plot\_China <- IPC\_RCAs\_Top4[IPC\_RCAs\_Top4$ctry\_code == country\_select[i],] %>%   
 arrange(Label, Period) %>% group\_by(Label) %>%   
 mutate( general = Total\_RCA\_2 == 1,   
 break\_in = Total\_RCA\_2 == 2,   
 break\_through = Total\_RCA\_2 == 3,   
 sustained\_general = general & lag(general, 1, default = FALSE),  
 sustained\_break\_in = break\_in & lag(break\_in, 1, default = FALSE),  
 sustained\_break\_through = break\_through & lag(break\_through, 1, default = FALSE)) %>%   
 ungroup()  
  
bar\_plot\_China <- bar\_plot\_China %>%   
 group\_by(Period) %>% summarise(`General case` = sum(general, na.rm = TRUE),  
 `Break-through case` = sum(break\_in, na.rm = TRUE),  
 `Break-in case` = sum(break\_through, na.rm = TRUE),  
 `Sustained General case` = sum(sustained\_general, na.rm = TRUE),  
 `Sustained break-through case` = sum(sustained\_break\_in, na.rm = TRUE),  
 `Sustained break-in case` = sum(sustained\_break\_through, na.rm = TRUE),  
 .groups = "drop") %>% arrange(Period)  
  
plot\_long\_China <- bar\_plot\_China |> rename(Period = Period) |>  
 pivot\_longer(cols= -Period,names\_to= "Indicator",values\_to = "Count")  
  
#order labels  
plot\_long\_China$Indicator <- factor(plot\_long\_China$Indicator, levels = rev(c("General case", "Break-through case", "Break-in case",   
 "Sustained General case", "Sustained break-through case", "Sustained break-in case")))  
plot\_long\_China$Period <- factor(plot\_long\_China$Period, levels = c("2004-2018", "1989-2003", "1974-1988"))  
  
legend\_order <- c(  
 "General case", "Break-through case", "Break-in case",  
 "Sustained General case", "Sustained break-through case", "Sustained break-in case"  
)  
  
 ggplot(plot\_long\_China, aes(x = factor(Period),y = Count, fill = Indicator)) +  
 geom\_col(position = position\_dodge(width = .8), width = .7) +  
 scale\_fill\_manual(values = c("General case" = "#FF3300",  
 "Sustained General case" = "#993333",  
 "Break-in case" = "#009900", #3399FF  
 "Sustained break-in case" = "#006633", #3333CC  
 "Break-through case" = "#3399FF", #009900  
 "Sustained break-through case" = "#3333CC"),  
 breaks = legend\_order) + #006633  
 guides(fill = guide\_legend(nrow = 2, byrow = TRUE)) +  
 labs(x = "Interval",y = "Number of cases", fill = NULL, title = NULL)+  
 ggtitle("Summary of specialisations China") +  
 theme\_classic(base\_size = 11) + theme(legend.position = "bottom")+ coord\_flip()

 The code is the basically the same for the remaining countries in interval. The countries and intervals are selected by varying the variables i (for countries), and p (for intervals). The figures are generated and combined by country using the custom function “multiplot”. The files are saved at “Files\_created\_with\_the\_code/figures/Figure\_5\_Specialisations\_techn\_space\_3\_periods\_4\_countries\_d\_China.jpg”, “Files\_created\_with\_the\_code/figures/Figure\_5\_Specialisations\_techn\_space\_3\_periods\_4\_countries\_b\_USA.jpg”, “Files\_created\_with\_the\_code/figures/Figure\_5\_Specialisations\_techn\_space\_3\_periods\_4\_countries\_a\_Japan.jpg”, and “Files\_created\_with\_the\_code/figures/Figure\_5\_Specialisations\_techn\_space\_3\_periods\_4\_countries\_c\_SouthKorea.jpg”.

### 1.3.2. AI-specific technological space (ATS)

Next, we create the ATS. We follow very similar steps: load the AI data, separate the patents that are specific to each interval, calculate the network and it’s coordinates, and plot one technological space per interval. The difference now is that the ATS is dynamic, meaning that we calculate the network every time for each interval. Starting with the first interval, the calculated degree for the top 10 most connected codes is:

g\_tech\_AI %N>% arrange(desc(dgr)) %>% as\_tibble() %>% slice(1:10)

## # A tibble: 10 × 5  
## name sector field\_name Category dgr  
## <chr> <chr> <chr> <chr> <dbl>  
## 1 12 Instruments Control AI-core fields 1   
## 2 6 Electrical engineering Computer technology AI-core fields 0.987  
## 3 7 Electrical engineering IT methods for management AI-core fields 0.861  
## 4 10 Instruments Measurement AI-core fields 0.731  
## 5 25 Mechanical engineering Handling Surrounding fie… 0.534  
## 6 26 Mechanical engineering Machine tools Other 0.523  
## 7 8 Electrical engineering Semiconductors Other 0.473  
## 8 32 Mechanical engineering Transport Other 0.447  
## 9 27 Mechanical engineering Engines, pumps, turbines Other 0.419  
## 10 29 Mechanical engineering Other special machines Other 0.417

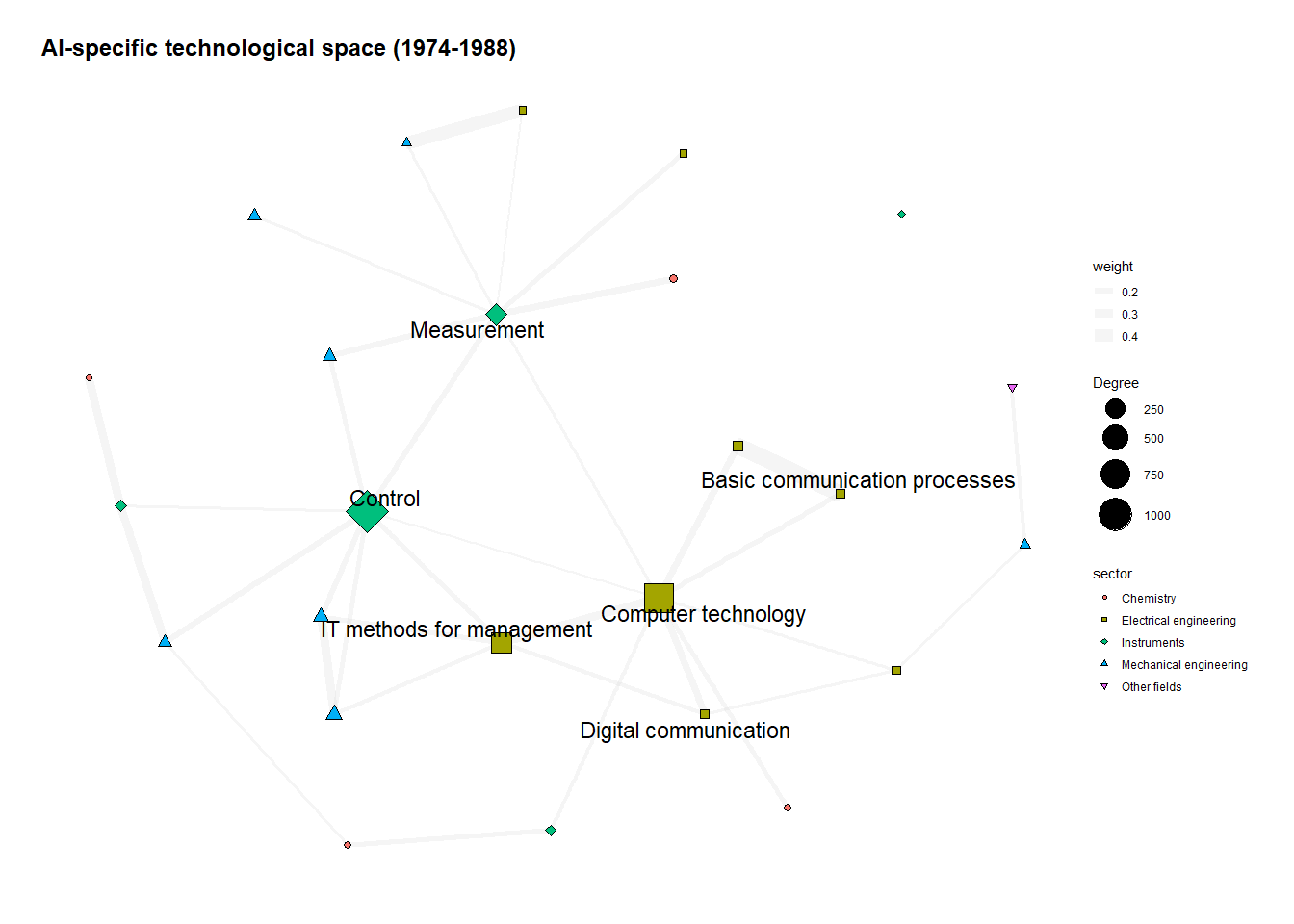
The dataset with the AI-specific specialisations (named AI\_RCA), saved previously in the file “Specializations\_All\_periods\_IPC.csv” (together with the specialisations of countries), looks like this:

head(AI\_RCA)

## techn\_field\_nr RCA\_AI\_Period Period\_sim Binary  
## 1 1 0.05934926 1 0  
## 2 2 0.08535377 1 0  
## 3 3 0.33606440 1 0  
## 4 4 1.09784935 1 1  
## 5 5 1.64528254 1 1  
## 6 6 16.64849577 1 1

Where “RCA\_AI\_Period” refers to the specific RTA of each code for each interval (which in turn is shown in the column Period\_sim). We then plot the ATS for the first interval:

AI\_RCA1 <- AI\_RCA[AI\_RCA$Period\_sim == 1,]  
p=1  
 g\_tech\_AI %N>%  
 left\_join(AI\_RCA1 %>% filter(Period\_sim == p), by = c("name" = "techn\_field\_nr")) %>%  
 ggraph(layout = coords\_tech\_AI) +   
 geom\_edge\_link(aes(width = weight), alpha = 0.2, colour = "#CCCCCC") +  
 geom\_node\_point(aes(fill = sector, size = 1000^dgr, shape= sector)) +  
 scale\_shape\_manual(values=c(21, 22, 23, 24, 25)) + labs(color = "RCA")+ scale\_size("Degree", range = c(2, 12)) +  
 geom\_node\_text(aes(filter=Binary > .99, label = field\_name), size = 6, repel = TRUE) +  
 theme\_graph(base\_family = "sans") + guides(colour = guide\_legend(override.aes = list(size=5)))+  
 ggtitle("AI-specific technological space (1974-1988)") #



We do the same for the 2 other intervals, and combine the three figures again using the multiplot custom function. The resulting figure is saved at “Files\_created\_with\_the\_code/figures/Figure\_2\_ATS\_and\_AI\_core\_technologies\_3\_intervals.jpg”

# 2. Other figures

Next, we create the 3 remaining figures shown in the paper (namely Figures 1, 6, and 7).

## 2.1. Share of Break-in specialisations (Fig 6 and 7)

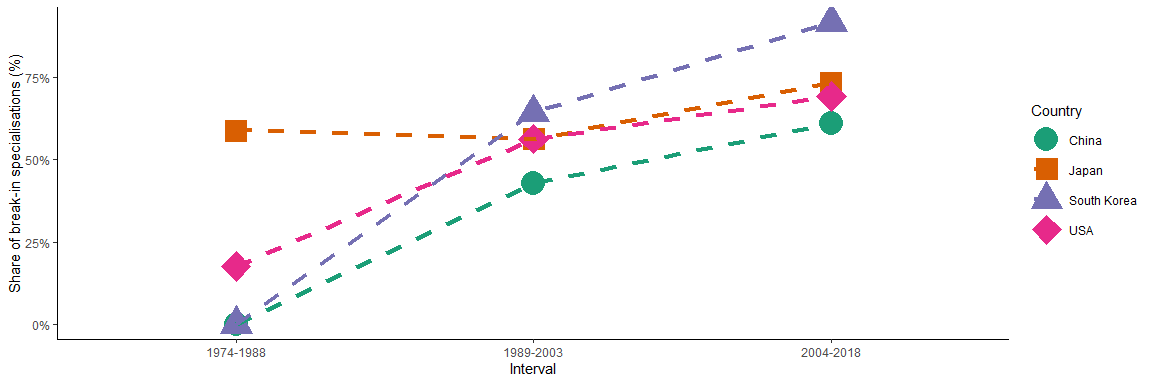
We create this figure by reading all the summary files we already separated before. We start by reading and combining files RCA\_4countries\_detailed.csv and Specializations\_All\_periods\_IPC.csv into a new file named IPC\_RCAs. Then, we summarize its main results, in the following way:

SummaryAllData<-distinct(IPC\_RCAs, ctry\_code, Period, .keep\_all = TRUE)   
colnames(SummaryAllData)[1] <- "Country"  
head(SummaryAllData)

## # A tibble: 6 × 20  
## Country techn\_field\_nr RCA\_Gen RCA\_AI Period Label Round\_general Round\_AI  
## <chr> <chr> <dbl> <dbl> <chr> <chr> <int> <int>  
## 1 China 1 0.781 0 1974-1… Elec… 0 0  
## 2 Japan 1 1.13 1.40 1974-1… Elec… 1 1  
## 3 South Korea 1 0.886 0 1974-1… Elec… 0 0  
## 4 USA 1 0.806 0 1974-1… Elec… 0 0  
## 5 China 1 0.686 2.09 1989-2… Elec… 0 1  
## 6 Japan 1 1.14 0.983 1989-2… Elec… 1 0  
## # ℹ 12 more variables: Total\_RCA <fct>, Period\_sim <dbl>, RCA\_AI\_Period <dbl>,  
## # Total\_RCA\_2 <dbl>, Coiciding <dbl>, justGeneral <dbl>, OnlyAI <dbl>,  
## # Share\_coinciding <dbl>, Share\_OnlyAI <dbl>, sum\_coinciding <dbl>,  
## # sum\_justGeneral <dbl>, sum\_OnlyAI <dbl>

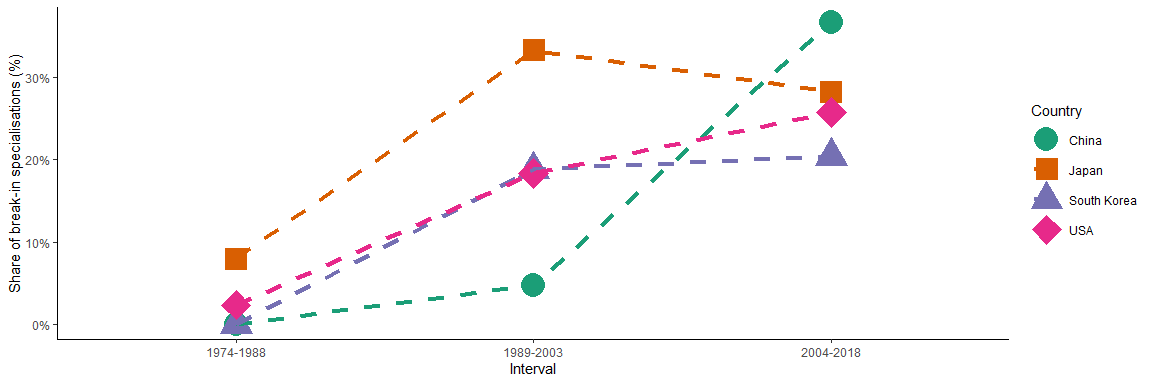
This file is all we need to create Figure 6, shown below (and saved in file “Files\_created\_with\_the\_code/figures/Figure\_6\_Share\_coinciding\_specialisations\_techn\_field.jpg”)

ggplot(data=SummaryAllData, aes(x=Period, y=Share\_coinciding, group=Country, shape = Country, color=Country)) +  
 geom\_point(aes(fill = Country), size=8) + scale\_shape\_manual(values=c(21, 22, 24, 23)) +  
 xlab("Interval") + ylab("Share of break-in specialisations (%)") +  
 theme\_classic() + geom\_line(aes(color=Country), linetype = "dashed", size=1.5)+  
 scale\_y\_continuous(labels = scales::percent) +  
 scale\_fill\_manual(values = c("#1B9E77", "#D95F02", "#7570B3", "#E7298A")) +  
 scale\_color\_manual(values = c("#1B9E77", "#D95F02", "#7570B3", "#E7298A"))



Next, we do the same for the subclass-based Figure 7, which is shown belown and saved as “Files\_created\_with\_the\_code/figures/Figure\_7\_Share\_coinciding\_specialisations\_subclass.jpg”.

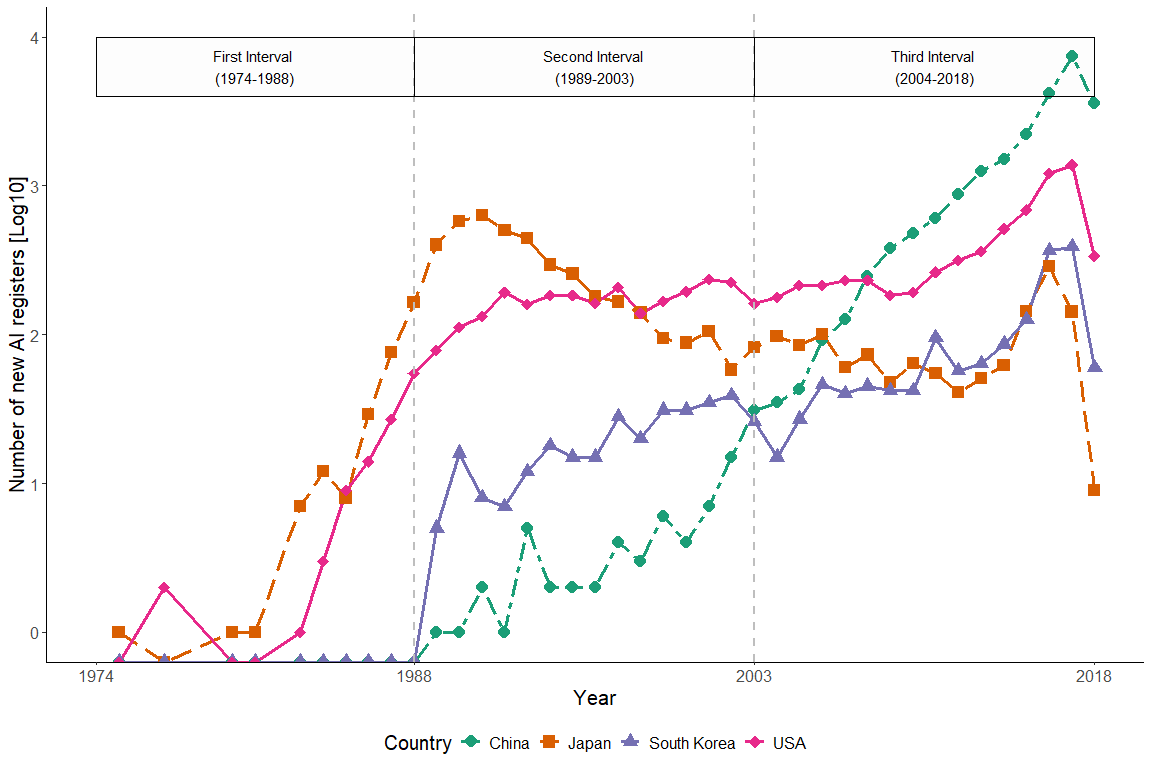
ggplot(data=SummaryAllData4dig, aes(x=Period, y=Share\_coinciding, group=Country, shape = Country, color=Country)) +  
 geom\_point(aes(fill = Country), size=8) +   
 scale\_shape\_manual(values=c(21, 22, 24, 23)) +  
 xlab("Interval") +  
 ylab("Share of break-in specialisations (%)") +  
 theme\_classic() +  
 geom\_line(aes(color=Country), linetype = "dashed", size=1.5)+  
 scale\_y\_continuous(labels = scales::percent) +  
 scale\_fill\_manual(values = c("#1B9E77", "#D95F02", "#7570B3", "#E7298A")) +  
 scale\_color\_manual(values = c("#1B9E77", "#D95F02", "#7570B3", "#E7298A"))



## 2.2. Log 10 of the number of AI patents by Japan, the USA, South Korea, and China

Next, we create Figure 1. Not much data is needed for this one (the only dataset needed is other\_files/IPCs\_AI.csv), and just simple commands used to process this data to get a summarized number. The produced figure is saved as “Files\_created\_with\_the\_code/figures/Figure\_1\_Log\_10\_AI\_patents\_per\_country.jpg”, and also shown below.

ggplot(data=test, aes(x=Year, y=log10(Number\_of\_AI\_patents), group=Country, colour=Country, shape=Country)) +  
 geom\_line(size=1.2, aes(linetype=Country)) +  
 geom\_point(size=4) + xlab("Year") + ylab("Number of new AI registers [Log10]") + theme\_classic() +  
 scale\_linetype\_manual(values=c("twodash", "longdash", "solid", "solid")) +  
 scale\_shape\_manual(values=c(16, 15, 17, 18)) + theme(legend.position="bottom") +  
 theme(text = element\_text(size = 15)) + scale\_y\_continuous(limits=c(0,4)) +   
 geom\_vline(data=test, aes(xintercept=c(1988), colour=Period), linetype="dashed", size=1, color = "grey") +   
 geom\_vline(data=test, aes(xintercept=c(2003), colour=Period), linetype="dashed", size=1, color = "grey") +   
 scale\_x\_continuous(breaks = c(1974, 1988, 2003, 2018), limits=c(1974, 2018)) + scale\_color\_brewer(palette="Dark2") +   
 annotate("rect", xmin = 1974, xmax=1988, ymin = 3.6, ymax = 4, alpha = .01, color = "black") +  
 annotate("text", x = 1981, y = 3.8, label = c("First Interval \n(1974-1988)"), size=4)+  
 annotate("rect", xmin = 1988, xmax=2003, ymin = 3.6, ymax = 4, alpha = .01, color = "black") +  
 annotate("text", x = 1996, y = 3.8, label = c("Second Interval \n(1989-2003)"), size=4) +  
 annotate("rect", xmin = 2003, xmax=2018, ymin = 3.6, ymax = 4, alpha = .01, color = "black") +  
 annotate("text", x = 2011, y = 3.8, label = c("Third Interval \n(2004-2018)"), size=4)



# 3. Permutations for technological fields

## 3.1. Permutate the AI dataset

Load the raw data, separate the interval (the first one for this example, from 1974 to 1988), and calculate the fractional count of each country in each considered field as done before.

The resulting count of techn\_fields per country is:

head(region\_tech\_fields\_1\_df)

## # A tibble: 6 × 3  
## ctry\_code techn\_field\_nr n\_tech\_reg  
## <chr> <int> <dbl>  
## 1 AD 20 1  
## 2 AD 24 1  
## 3 AD 28 3  
## 4 AD 32 1  
## 5 AD 33 1  
## 6 AD 34 3

Now we load the AI patents, select the ones from the first interval, define target countries (the four main ones), and define the number of permutations. To avoid that it takes it too long, I’ll use 10 permutations (num\_permutations = 10; in the paper, this number is set to 1000).

list\_of\_permuted\_dfs <- vector("list", length = num\_permutations)  
  
for (p in 1:num\_permutations) {  
 if (p %% 100 == 0) print(paste("Permutation number:", p)) # Progress indicator  
   
 # This dataframe will hold the permuted AI patents for target countries ONLY for THIS iteration  
 permuted\_ai\_for\_target\_countries\_iter <- data.frame()  
   
 for (country in target\_countries) {  
 # 1. Identify and Count ACTUAL AI patents for the current country from the original AI dataset  
 actual\_ai\_appln\_ids\_country <- ai\_patents\_period\_1\_df %>%  
 filter(ctry\_code == country) %>%  
 distinct(appln\_id) %>%  
 pull(appln\_id)  
   
 n\_ai\_country <- length(actual\_ai\_appln\_ids\_country)  
   
 if (n\_ai\_country == 0) {  
 # print(paste("No AI patents found for", country, "in original AI data. Skipping for perm", p))  
 next # Skip to the next country if no AI patents to replace  
 }  
   
 # 2. Prepare the pool of ALL patents for the current country from the general dataset  
 country\_all\_patents\_pool <- ipc\_all\_patents\_first\_period\_df %>%  
 filter(ctry\_code == country) %>%  
 distinct(appln\_id)  
   
 if (nrow(country\_all\_patents\_pool) == 0) {  
 # print(paste("No patents in general pool for", country, ". Skipping for perm", p))  
 next  
 }  
   
 # Handle cases where the pool is smaller than the number of AI patents to sample  
 # This is unlikely if ipc\_all\_patents\_first\_period\_df is complete, but good for robustness  
 sample\_size <- min(n\_ai\_country, nrow(country\_all\_patents\_pool))  
 replace\_sampling <- FALSE  
 if (n\_ai\_country > nrow(country\_all\_patents\_pool)) {  
 # print(paste("Warning: For country", country, "in perm", p,  
 # "not enough unique patents in pool. Sampling", nrow(country\_all\_patents\_pool),  
 # "instead of", n\_ai\_country, "OR consider sampling with replacement."))  
 # Decide: either sample fewer (as done with min()), or sample with replacement.  
 # If sampling with replacement is desired:  
 # sample\_size <- n\_ai\_country  
 # replace\_sampling <- TRUE  
 # For now, we sample up to the available pool size without replacement.  
 # Or, if strict adherence to n\_ai\_country is needed and pool is too small WITH replace=FALSE:  
 if(nrow(country\_all\_patents\_pool) < n\_ai\_country && !replace\_sampling){  
 # print(paste("Strict N\_AI needed, but pool too small for", country, "in perm", p, ". Skipping country for this perm."))  
 next # Skip this country for this permutation if not enough patents  
 }  
 }  
   
   
 # 3. Randomly select an equivalent number of unique appln\_ids from this country's general pool  
 random\_appln\_ids\_country <- sample(country\_all\_patents\_pool$appln\_id,  
 size = sample\_size, # Use adjusted sample\_size  
 replace = replace\_sampling) # Use replace\_sampling flag  
   
 # 4. Get all rows for these randomly selected patents from the ipc\_all\_patents\_first\_period\_df  
 randomly\_selected\_patents\_df\_country <- ipc\_all\_patents\_first\_period\_df %>%  
 filter(appln\_id %in% random\_appln\_ids\_country & ctry\_code == country)  
   
 # 5. Add these randomly selected patents for the current country to the iteration's df  
 if (nrow(randomly\_selected\_patents\_df\_country) > 0) {  
 permuted\_ai\_for\_target\_countries\_iter <- bind\_rows(  
 permuted\_ai\_for\_target\_countries\_iter,  
 randomly\_selected\_patents\_df\_country  
 )  
 }  
 } # End of country loop  
   
 # Add the permutation number to all rows of this iteration's dataframe  
 if (nrow(permuted\_ai\_for\_target\_countries\_iter) > 0) {  
 permuted\_ai\_for\_target\_countries\_iter$permutation\_number <- p  
 }  
   
 # Store the dataframe for this iteration in the list  
 list\_of\_permuted\_dfs[[p]] <- permuted\_ai\_for\_target\_countries\_iter  
   
} # End of permutation loop  
  
# Combine all permuted dataframes from the list into one large dataframe  
final\_permuted\_dataset <- bind\_rows(list\_of\_permuted\_dfs)

The resulting permuted dataset looks like this for the 5 initial and 5 last lines:

head(final\_permuted\_dataset)

## appln\_id ctry\_code techn\_field\_nr permutation\_number  
## 1 25158951 JP 10 1  
## 2 25207766 JP 2 1  
## 3 25207766 JP 2 1  
## 4 25207766 JP 2 1  
## 5 25256648 JP 1 1  
## 6 25256648 JP 1 1

tail(final\_permuted\_dataset)

## appln\_id ctry\_code techn\_field\_nr permutation\_number  
## 11620 47080144 US 2 10  
## 11621 48456520 US 6 10  
## 11622 47080144 US 2 10  
## 11623 52219554 US 14 10  
## 11624 52219554 US 23 10  
## 11625 48823043 US 3 10

Or, in summary, it looks like this (please don’t forget that just Japan and the USA from the 4 selected countries had patents in the first interval):

final\_permuted\_dataset %>%  
 filter(permutation\_number <= 5) %>%  
 group\_by(permutation\_number, ctry\_code) %>%  
 summarise(unique\_appln\_ids = n\_distinct(appln\_id), .groups = 'drop') %>%  
 print(n=20)

## # A tibble: 10 × 3  
## permutation\_number ctry\_code unique\_appln\_ids  
## <int> <chr> <int>  
## 1 1 JP 307  
## 2 1 US 107  
## 3 2 JP 307  
## 4 2 US 107  
## 5 3 JP 307  
## 6 3 US 107  
## 7 4 JP 307  
## 8 4 US 107  
## 9 5 JP 307  
## 10 5 US 107

Next, we separate the list of NOT-targeted countries with AI patents, create a list to hold the replicated dataframes, and loop these countries according to the number of permutations selected (10 in this example), so that we also have these ‘extra’ AI-players on every permutation.

The resulting dataset looks like this for the first and last 6 observations

head(replicated\_not\_selected\_ai\_final)

## appln\_id ctry\_code permutation\_number  
## <int> <char> <int>  
## 1: 16723353 FR 0  
## 2: 16723353 FR 0  
## 3: 16723353 FR 0  
## 4: 36147193 IE 0  
## 5: 36147193 IE 0  
## 6: 36147193 IE 0

tail(replicated\_not\_selected\_ai\_final)

## appln\_id ctry\_code permutation\_number  
## <int> <char> <int>  
## 1: 16709578 DE 10  
## 2: 16709578 DE 10  
## 3: 16709578 DE 10  
## 4: 10332913 DE 10  
## 5: 10332913 DE 10  
## 6: 10332913 DE 10

We merge this back into the “target” dataset, name the country\_code of it as “AI\_pat” to be able to trace it back, and merge everything.

The result looks like this:

table(final\_permuted\_dataset$permutation\_number)

##   
## 0 1 2 3 4 5 6 7 8 9 10   
## 1418 1205 1135 1243 1153 1194 1171 1134 1143 1140 1156

## 3.2. Calculate AI-specific specialisations

Now we calculate the AI-specific specialisations based on these Permuted dataset:

list\_of\_rca\_dfs <- region\_tech\_fields\_perm\_df %>%  
 group\_by(permutation\_number) %>%  
 group\_split() %>% # This splits the df into a list of dfs, one for each permutation  
 purrr::map(~{  
 current\_permutation\_number <- unique(.x$permutation\_number)  
 print(paste("Processing RCA for permutation\_number:", current\_permutation\_number))  
   
 # Matrix creation for the current permutation's data  
 mat\_reg\_tech\_perm\_AI <- .x %>%  
 select(-permutation\_number) %>% # Temporarily remove for pivot if it causes issues  
 arrange(techn\_field\_nr, ctry\_code) %>%  
 pivot\_wider(names\_from = techn\_field\_nr,  
 values\_from = n\_tech\_reg,  
 values\_fill = 0) # Changed from list(n\_tech\_reg = 0) for simplicity  
   
 # Check if ctry\_code column exists and is not empty  
 if (!"ctry\_code" %in% names(mat\_reg\_tech\_perm\_AI) || nrow(mat\_reg\_tech\_perm\_AI) == 0 || all(is.na(mat\_reg\_tech\_perm\_AI$ctry\_code))) {  
 print(paste("Skipping permutation", current\_permutation\_number, "due to missing ctry\_code or empty data after pivot."))  
 return(NULL) # Return NULL or an empty tibble  
 }  
   
 # Check for duplicate ctry\_codes which would prevent rownames\_to\_column  
 if (any(duplicated(mat\_reg\_tech\_perm\_AI$ctry\_code))) {  
 print(paste("Warning: Duplicate ctry\_code found for permutation", current\_permutation\_number, ". Aggregating or handling needed."))  
 return(tibble(permutation\_number = current\_permutation\_number, error="duplicate ctry\_code"))  
 }  
   
   
 mat\_reg\_tech\_perm\_AI <- mat\_reg\_tech\_perm\_AI %>%  
 remove\_rownames() %>%  
 column\_to\_rownames(var = "ctry\_code") %>%  
 as.matrix() %>% round()# No rounding here, location\_quotient might prefer raw numbers  
   
 # RCA calculation  
 # Ensure matrix is suitable (e.g., no NA/NaN/Inf that location\_quotient can't handle)  
 if (nrow(mat\_reg\_tech\_perm\_AI) == 0 || ncol(mat\_reg\_tech\_perm\_AI) == 0) {  
 print(paste("Skipping RCA for permutation", current\_permutation\_number, "due to empty matrix."))  
 return(NULL)  
 }  
   
 # Ensure there are at least two columns for location\_quotient (ctry\_code was one)  
 if (ncol(mat\_reg\_tech\_perm\_AI) < 1) { # If only ctry\_code was present and now it's rownames  
 print(paste("Skipping RCA for permutation", current\_permutation\_number, "due to insufficient columns in matrix."))  
 return(NULL)  
 }  
   
   
 # Check for all zero rows/columns if location\_quotient is sensitive  
 # For example, if a row sum is 0, RCA might be NaN or Inf.  
 # The location\_quotient function might handle this, or you might need pre-filtering.  
   
 rca\_results\_perm <- tryCatch({  
 mat\_reg\_tech\_perm\_AI %>%  
 location\_quotient(binary = FALSE) %>%   
 as.data.frame() %>%  
 rownames\_to\_column("ctry\_code") %>%  
 as\_tibble() %>%  
 gather(key = "techn\_field\_nr", value = "RCA", -ctry\_code) %>%  
 arrange(ctry\_code, techn\_field\_nr) %>%  
 mutate(permutation\_number = current\_permutation\_number) # Add back permutation number  
 }, error = function(e) {  
 print(paste("Error in location\_quotient for permutation", current\_permutation\_number, ":", e$message))  
 return(tibble(permutation\_number = current\_permutation\_number, ctry\_code=NA, techn\_field\_nr=NA, RCA=NA, error\_message = e$message)) # Return an empty or error-marked tibble  
 })  
   
 return(rca\_results\_perm)  
 })

## [1] "Processing RCA for permutation\_number: 0"  
## [1] "Processing RCA for permutation\_number: 1"  
## [1] "Processing RCA for permutation\_number: 2"  
## [1] "Processing RCA for permutation\_number: 3"  
## [1] "Processing RCA for permutation\_number: 4"  
## [1] "Processing RCA for permutation\_number: 5"  
## [1] "Processing RCA for permutation\_number: 6"  
## [1] "Processing RCA for permutation\_number: 7"  
## [1] "Processing RCA for permutation\_number: 8"  
## [1] "Processing RCA for permutation\_number: 9"  
## [1] "Processing RCA for permutation\_number: 10"

# Combine the list of RCA dataframes into one final dataframe  
final\_rca\_all\_permutations\_df <- bind\_rows(list\_of\_rca\_dfs)

The new AI-specific RTAs look like this:

head(final\_rca\_all\_permutations\_df)

## # A tibble: 6 × 4  
## ctry\_code techn\_field\_nr RCA permutation\_number  
## <chr> <chr> <dbl> <dbl>  
## 1 AT 1 0 0  
## 2 AT 10 0 0  
## 3 AT 11 0 0  
## 4 AT 12 0 0  
## 5 AT 13 0 0  
## 6 AT 17 0 0

tail(final\_rca\_all\_permutations\_df)

## # A tibble: 6 × 4  
## ctry\_code techn\_field\_nr RCA permutation\_number  
## <chr> <chr> <dbl> <dbl>  
## 1 US 4 0 10  
## 2 US 5 0 10  
## 3 US 6 1.02 10  
## 4 US 7 0 10  
## 5 US 8 0.263 10  
## 6 US 9 0.892 10

This file (or better said, the relevant file with 1000 permutations) is saved as “final\_rca\_all\_permutations\_df\_1st\_Period.csv”. The other two intervals are saved with similar names, i.e., “final\_rca\_all\_permutations\_df\_2nd\_Period.csv” and “final\_rca\_all\_permutations\_df\_3rd\_Period.csv”, in Files\_created\_with\_the\_code/data/files\_code\_Fields\_analysis/robustness/

# 4. Econometrics

Read the files calculated in the “3.Robustness” code, and calculate the first three models.

The dataset looks like this

head(regression\_data)

## ctry\_code rel\_density period no\_specialization general\_specialization  
## 1 JP 57 1974-1978 15 19  
## 2 KR 38 1974-1978 22 13  
## 3 US 50 1974-1978 19 15  
## 4 CN 35 1974-1978 24 11  
## 5 CN 32 1979-1983 23 12  
## 6 JP 56 1979-1983 17 16  
## ai\_specific\_specialization coinciding\_specialization actual\_share\_coinciding  
## 1 0 1 0.0500000  
## 2 0 0 0.0000000  
## 3 0 1 0.0625000  
## 4 0 0 0.0000000  
## 5 0 0 0.0000000  
## 6 0 2 0.1111111  
## actual\_share\_round\_ai actual\_share\_round\_general actual\_persistent\_coinciding  
## 1 0.02857143 0.5714286 0  
## 2 0.00000000 0.3714286 0  
## 3 0.02857143 0.4571429 0  
## 4 0.00000000 0.3142857 0  
## 5 0.00000000 0.3428571 0  
## 6 0.05714286 0.5142857 1  
## actual\_persistent\_just\_general actual\_persistent\_just\_ai  
## 1 0 0  
## 2 0 0  
## 3 0 0  
## 4 0 0  
## 5 5 0  
## 6 15 0  
## actual\_n\_persistent\_round\_ai actual\_n\_ai\_prev\_coinciding  
## 1 0 0  
## 2 0 0  
## 3 0 0  
## 4 0 0  
## 5 0 0  
## 6 1 0  
## actual\_n\_coinciding\_prev\_ai actual\_n\_ai\_prev\_gen  
## 1 0 0  
## 2 0 0  
## 3 0 0  
## 4 0 0  
## 5 0 0  
## 6 0 0  
## actual\_n\_persistent\_core\_fields actual\_n\_persistent\_not\_core\_fields  
## 1 0 0  
## 2 0 0  
## 3 0 0  
## 4 0 0  
## 5 0 0  
## 6 1 0  
## actual\_n\_persistent\_coin\_core\_fields actual\_n\_persistent\_coin\_not\_core\_fields  
## 1 0 0  
## 2 0 0  
## 3 0 0  
## 4 0 0  
## 5 0 0  
## 6 1 0  
## actual\_ai\_core\_fields actual\_ai\_not\_core\_fields actual\_persistent\_general\_all  
## 1 1 0 0  
## 2 0 0 0  
## 3 0 1 0  
## 4 0 0 0  
## 5 0 0 5  
## 6 2 0 17  
## total\_general\_specializations double\_check total\_specializations  
## 1 20 35 20  
## 2 13 35 13  
## 3 16 35 16  
## 4 11 35 11  
## 5 12 35 12  
## 6 18 35 18  
## total\_AI\_specializations Share\_Coinciding Period  
## 1 1 0.0500000 1974-1978  
## 2 0 0.0000000 1974-1978  
## 3 1 0.0625000 1974-1978  
## 4 0 0.0000000 1974-1978  
## 5 0 0.0000000 1979-1983  
## 6 2 0.1111111 1979-1983

The results for the first three models are:

stargazer(model3, model1, model2, title="Effects on the share of break-ins", ci.level=0.95, single.row=TRUE, ci=F, type="text")

##   
## Effects on the share of break-ins  
## ===================================================================================================  
## Dependent variable:   
## ---------------------------------------------------------------------  
## Share\_Coinciding   
## (1) (2) (3)   
## ---------------------------------------------------------------------------------------------------  
## rel\_density 0.001 (0.005) 0.001 (0.006) -0.001 (0.006)   
## total\_general\_specializations -0.010 (0.016) 0.002 (0.018)   
## Period1979-1983 0.004 (0.132) -0.012 (0.074) -0.013 (0.075)   
## Period1984-1988 0.164 (0.132) -0.016 (0.079) -0.045 (0.084)   
## Period1989-1993 0.348\*\* (0.131) 0.019 (0.085) -0.005 (0.089)   
## Period1994-1998 0.405\*\*\* (0.131) 0.005 (0.093) -0.002 (0.095)   
## Period1999-2003 0.455\*\*\* (0.136) -0.007 (0.098) -0.034 (0.103)   
## Period2004-2008 0.535\*\*\* (0.132) 0.080 (0.095) 0.048 (0.100)   
## Period2009-2013 0.534\*\*\* (0.131) 0.065 (0.096) 0.042 (0.101)   
## Period2014-2018 0.666\*\*\* (0.132) 0.118 (0.105) 0.093 (0.110)   
## ctry\_codeJP -0.018 (0.055)   
## ctry\_codeKR 0.063 (0.054)   
## ctry\_codeUS 0.049 (0.051)   
## total\_AI\_specializations 0.036\*\*\* (0.005) 0.038\*\*\* (0.005)   
## Constant -0.036 (0.227) 0.100 (0.130) 0.017 (0.152)   
## ---------------------------------------------------------------------------------------------------  
## Observations 36 36 36   
## R2 0.673 0.905 0.915   
## Adjusted R2 0.560 0.862 0.859   
## Residual Std. Error 0.185 (df = 26) 0.104 (df = 24) 0.105 (df = 21)   
## F Statistic 5.958\*\*\* (df = 9; 26) 20.802\*\*\* (df = 11; 24) 16.181\*\*\* (df = 14; 21)  
## ===================================================================================================  
## Note: \*p<0.1; \*\*p<0.05; \*\*\*p<0.01

For the remaining 3 models, the results are:

stargazer(newmodel4, newmodel5, newmodel6, title="Effects on persisting specialisations", ci.level=0.95, single.row=TRUE, ci=F, type="text")

##   
## Effects on persisting specialisations  
## =========================================================================================================  
## Dependent variable:   
## ---------------------------------------------------------------------------  
## actual\_persistent\_coinciding actual\_n\_persistent\_round\_ai  
## (1) (2) (3)   
## ---------------------------------------------------------------------------------------------------------  
## rel\_density 0.015 (0.031) 0.039 (0.082) 0.006 (0.078)   
## Period1979-1983 -1.431 (1.171) 0.480 (1.092) -0.124 (1.044)   
## Period1984-1988 -1.901 (1.120) -0.941 (1.192) -0.628 (1.151)   
## Period1989-1993 -1.942 (1.330) 0.423 (1.258) -0.292 (1.201)   
## Period1994-1998 -1.263 (1.148) 1.854 (1.368) 0.158 (1.354)   
## Period1999-2003 -1.280 (1.234) 1.708 (1.477) -0.520 (1.447)   
## Period2004-2008 -1.812 (1.245) 0.919 (1.405) 0.807 (1.350)   
## Period2009-2013 0.012 (1.305) 3.546\*\* (1.420) -1.135 (1.524)   
## Period2014-2018 -0.420 (1.297) 2.861\* (1.572) -0.458 (1.601)   
## actual\_persistent\_general\_all 0.133\* (0.074)   
## actual\_n\_persistent\_round\_ai 0.525\*\*\* (0.070)   
## total\_general\_specializations 0.113 (0.239) -0.053 (0.228)   
## actual\_persistent\_coinciding 1.115\*\*\* (0.199)   
## coinciding\_specialization 0.413\* (0.226) -0.182 (0.231)   
## total\_AI\_specializations -0.067 (0.150) 0.297\*\* (0.143)   
## Constant -0.669 (1.464) -3.642\* (1.955) 0.482 (1.997)   
## ---------------------------------------------------------------------------------------------------------  
## Observations 36 36 36   
## R2 0.914 0.799 0.921   
## Adjusted R2 0.874 0.694 0.875   
## Residual Std. Error 0.981 (df = 24) 1.531 (df = 23) 1.458 (df = 22)   
## F Statistic 23.111\*\*\* (df = 11; 24) 7.606\*\*\* (df = 12; 23) 19.828\*\*\* (df = 13; 22)   
## =========================================================================================================  
## Note: \*p<0.1; \*\*p<0.05; \*\*\*p<0.01