Diagram

Description automatically generated

**iCite\_WOS**xx (see pipeline.py lines 1,030 – 1,041)**:**

This module combines records from the NIH RePORTER (columns ‘PMID’, ‘PubYr’, and ‘DHHS\_Grant’) [6], the NIH-supported iCite platform (columns ‘PMID’, ‘PubYr’, and ‘RCR’), for original research articles (as per iCite criteria) [7], and Clarivate™ Web of Science™ tool (columns ‘PMID’, ‘DT’, ‘FU’, and ‘FX’) for calendar years 2010 through 2019 (xx).

xx *Access to the Web of Science™ tool comes through a contract with Clarivate Analytics that forbids redistribution of their database. Therefore, itemized data for columns ‘DT’ (Document Types), ‘FU’ (Funding Organizations), and ‘FX’ (Funding Text) were redacted. Researchers who desire the raw data on which to run our analysis can obtain it via a paid subscription to Clarivate Analytics.*

Input datasets: BIBL2010, BIBL2011, BIBL2012, BIBL2013, BIBL2014, BIBL2015, BIBL2016, BIBL2017, BIBL2018, BIBL2019

Downstream module: RJD1

|  |  |
| --- | --- |
| **Input columns:** | **Output columns:** |
| ***BIBL2010, BIBL2011, BIBL2012, BIBL2013, BIBL2014, BIBL2015, BIBL2016, BIBL2017, BIBL2018, BIBL2019:***  PMID – PubMed Identifier  PubYr – Article’s publication year  RCR – Relative Citation Ratio [8]  DHHS\_Grant – List of awards that have acknowledged support of the published research article. The records include only awards provided by the US Federal funding entities annotated in the NIH RePORTER database.  DT – Document type  FU – List of funding organizations and awards supporting the article (may enumerate funders and awards beyond those included in the NIH RePORTER database)  FX – Funding text (narrative of the Acknowledgement section of the research article) | ***iCite\_WOS:***  PMID – PubMed Identifier  PubYr – Article’s publication year  RCR – Relative Citation Ratio [8]  DHHS\_Grant – List of awards that have acknowledged support of the published research article. The records include only awards provided by the US Federal funding entities annotated in the NIH RePORTER database.  DT – Document type  FU – List of funding organizations and awards supporting the article (may enumerate funders and awards beyond those included in the NIH RePORTER database)  FX – Funding text (narrative of the Acknowledgement section of the research article) |

Comments:

**Lines 1 – 2:** Define the input columns

**Lines 3 – 11:** Append bibliography data for years 2011 – 2019.

●●●

**RJD1** (see pipeline.py lines 718 – 895)**:**

This module calculated the number of funding sources for each publication using data from iCite and the Web of Science.

Input dataset: iCite\_WOS

Downstream modules: RJD2, RJD4, INV\_COIN\_CALC, AWARD\_COIN\_CALC

|  |  |
| --- | --- |
| **Input columns:** | **Output columns:** |
| ***iCite\_WOS:***  PMID – PubMed Identifier  PubYr – Article’s publication year  RCR – Relative Citation Ratio [8]  DHHS\_Grant – List of awards that have acknowledged support of the published research article. The records include only awards provided by the US Federal funding entities annotated in the NIH RePORTER database.  DT – Document type  FU – List of funding organizations and awards supporting the article (may enumerate funders and awards beyond those included in the NIH RePORTER database)  FX – Funding text (narrative of the Acknowledgement section of the research article) | ***RJD1:***  PMID – Article’s PubMed Identifier  PubYr – Article’s publication year  DHHS\_GrantN – Number of acknowledged RePORTER grants  Pub – publication count  RCR – Relative Citation Ratio [8]  PubPerHHSGr\_FRAC – Fractional impact of an article based on one publication count divided by the number of acknowledged RePORTER awards  PubPerALLGr\_FRAC - Fractional impact of an article based on one publication count divided by the number of all acknowledged awards  RCRperHHSGr\_FRAC - Fractional impact of an article based on article’s RCR value divided by the number of acknowledged RePORTER awards  RCRperALLGr\_FRAC – Fractional impact of an article based on article’s RCR value divided by the number of all acknowledged awards |

Comments:  
**Line 1:** Import PySpark functions  
**Lines 2 - 3:** Define the original dataframes.  
**Lines 4- 6:** Replace null values with "0"s in columns "DT", "FU", and "RCR".  
**Line 7:** This operation converts string-type column into an array-type column with selected delimiter as second argument. Here, in column DHHS\_Grant, there are semicolons present in the original strings. This operation allows F.explode function to be performed in the next step. Note that an array-type column cannot be modified by regexp\_replace and other editing functions designed for string-type columns. F.split takes Java regex as second argument.  
**Line 8:** F.explode function takes in an array or a map and as an input, and outputs the elements of the array (map) into separate rows. The function .alias may be used to rename multiple columns, e. g., df = df.select(col("name").alias("new\_name").  
**Line 9:** Select parts of grant numbers necessary for the next steps. In PySpark, the sql.substring(str, pos, len) function has three arguments: str (i.e., the name of the column from which the original string will be taken), pos (designating position from which the counting of the characters to be retained will start; preceded by ‘-’ (minus) if counting starts from the right), and len (designating the length of the substring to be retained). This function is used to extract the substring from a DataFrame string column. The position is not zero based, but 1 based index. Here ‘-8', ‘-’ (minus) means the count starts from the right side of the string (e, g., for string {L-11}{6-10}{0-9}{M-8}{D-7}{0-6}{0-5}{8-4}{3-3}{2-2}{9-1}). Here, starting from position ‘-8', the next eight characters should be taken into column 'DHHS\_GrantT' from column 'DHHS\_GrantE’ to make it identical to the FUT column grant number after execution of the next steps.  
**Line 10:** Drop rows with duplicate values in selected columns. Here the values in "PMID" and "DHHS\_GrantT" columns in one row should be identical to values in "PMID" and "DHHS\_GrantT" in another row. This operation removes duplicative records for grants that change activity codes (e,g, R43 and R44, or R41 and R42).  
**Line 11:** Group rows by PMIDs; count the number of rows in each PMID and record this count in column “DHHS\_GrantN”; aggregate exploded grant numbers in column “DHHS\_GrantT”. PySpark groupBy() function is used to collect the identical data into groups on DataFrame and perform aggregate functions on the grouped data, e.g., sum() - Returns the total for values for each group; agg() function calculates more than one aggregate at a time. Two functions, groupBy() and agg(), are typically used in tandem, but agg(), can be used in a dataset without groupBy(), but typically it is not useful. PySpark SQL collect\_list() and collect\_set() functions are used to create an array type column on DataFrame by merging rows, typically after group by or window partitions. F.count is used to count values or strings in a row of an array-type column.  
**Lines 12 - 13:** In preparation for line 14 substitution of commas (',') with '@', to prevent interference with the existing '@'s, remove all '@' characters from columns FU and FX using regexp\_replace function.  
**Line 14:** Create FU1 column as a derivative of FU column and substitute all commas (“,”) there for “@”s to prevent split in later operations of comma-containing strings and substrings that do not enumerate grant numbers (such substrings are located outside of the square brackets). Also, for replacement of single characters, sql translate function can be used, e.g., df=df.withColumn('FX1',F.translate('FX1','A','B')) will substitute all 'A's for 'B's.  
**Line 15:** Separate elements of the FU1 array into separate rows by executing F.explode and placing these elements in separate rows of column FU2.  
**Line 16:** Remove occasional '<' and '>' characters from column FU2.  
**Line 17:** Substitute all square brackets for angle brackets in column FU2. Angle brackets are easier to manipulate because they are not metacharacters in Java and Python. Square brackets are metacharacters in Python. To negate square brackets as metacharacters they should be preceded by backslash. However, backslash is a metacharacter in Java, which is intermediary in PySpark translation from Python to Apache Spark, and thus should be negated by another backslash in PySpark.  
**Line 18:** Remove part of the string that is not in brackets. This operation would get rid of the duplicative funding agency name information and retain only grant numbers that are provided by this agency. Originally, Clarivate Web of Science database put this numbers in square brackets that follow the agency's name in the same “phrase”, i. e., in-between semicolons. Here regex finds strings/cells that consists of characters that are not an open/closed angle brackets (as shown by caret '^' in group 1) and also has a sub-string in-between angle brackets at the end of the entire string (group 2). The code will retain only the second part of the string that is in angle brackets (group 2). The metacharacters '.' designates any character, '*' - any number of any characters designated by the preceding character, and '?' after '*' limits this inclusion to the minimum.  
**Line 19:** Create a new column FU3 and place there the strings from column FU2 that are not delineated by angle brackets. Separation of angle bracket-delineated strings and the strings without angle brackets is necessary to remove '@'s (originally ',' ) from strings without angle brackets so that these strings will not be misread by the explode function and, as a result, counted as several separate grants.  
**Line 20:** Create column FU4, in which only strings delineated by angle brackets will be present. This is opposite to column FU3 in which only strings not delineated by angle brackets are present. The command below takes the content of column FU2, eliminates non-angle bracket-delineated strings, and places the result in column FU4.  
**Line 21:** Remove '@'s (former ',') from column FU3 containing strings without angle brackets.  
**Line 22:** FU3 and FU4, are concatenated to create column FU5, which is a re-creation of column FU2, except that strings outside of angle brackets do not contain '@'s (former ','s).  
**Line 23:** In preparation for aggregation of the exploded column FU5, this operation deletes angle brackets and substitutes '@'s for ','s.  
**Line 24:** Strings in different rows of column FU5 corresponding to the same PMID are aggregated and placed in array-type new column FU5. PySpark groupBy() function is used to collect the identical data into groups on dataframe and perform aggregate functions on the grouped data.  
**Line 25:** An array-type column FU5 created in the previous step is transformed into a string-type column with semicolon (';') as a divider to allow transformations to be performed in the next steps.  
**Line 26:** To create uniform divider (';') between sub-strings in column FU5, substitute all commas (',') for semicolons (';').  
**Lines 27 - 28:** Array-type columns DHHS\_Grant and DHHS\_GrantT created in the previous steps are transformed into a string-type column with semicolon (';') as a divider to allow transformations to be performed in the next steps.  
**Line 29:** Substitute all remaining brackets for angle brackets and move the results from column FU5 into column FUT.  
**Line 30 - 92:** In "FUT" column: (1) Remove standard words and phrases; (2) Remove spaces between fragments of the same grant numbers; (3) Remove words that do not contain numbers and capital letters; (4) Remove single capital letters (may be followed by a period); (5) Remove flanking parentheses.  
**Line 93:** Remove empty spaces between semicolons and text.  
**Lines 94 - 95:** Exploding column DHHS\_GrantT to compare with FU funders list in the next steps.  
**Line 96:** Truncating column DHHS\_GrantT1 to a substring of four character long, starting from the right and moving the resulting substring to column DHHS\_GrantT2.  
**Line 97:** Duplicate column DHHS\_GrantT2 to create column DHHS\_GrantT3. This is done to create two patterns of comparison between the FUT and DHHS\_GrantT strings: one for PMID records with fewer elements in column DHHS\_Grant and another for records with more elements in column DHHS\_Grant (>9).  
**Line 98:** To find similarities in award numbers that may be mistyped or truncated, split 4-digit strings in column DHHS\_GrantT2 into 2 overlapping 3-digit strings separated by semicolons.  
**Line 99:** Substitute content in column DHHS\_GrantT2 with content in column DHHS\_GrantT3 if the number in column DHHS\_GrantN is more than 9. The resulting column DHHS\_GrantT2 will be compared with FU content.  
**Lines 100 - 101:** Split column DHHS\_GrantT2 into an array column and explode into column DHHS\_GrantT3.  
**Lines 102 - 103:** In preparation for comparing with columns DHHS\_GrantT2 and DHHS\_GrantT3, split column FUT into an array column and explode into column FUTEX1.  
**Line 104:** Trim column FUTEX1 (remove leading and trailing empty spaces).  
**Line 105:** Interspace standard grant numbers with ‘%’s to protect from modifications in the next steps.

**Line 106:** Duplicate column FUTEX2 to create column FUTEX3 to modify the grant numbers that can be matched to standard ones in column DHHS\_GrantT (in column FUTEX3) while preserving the original grant name/number format in column FUTEX2.

**Lines 107 - 113:** Remove potential suffixes in likely NIH grant numbers in column FUTEX2 defining: (line 107) – 4 last characters in 10-character numbers where all characters are digits except the one before last that can be A or S; (line 108) – last 2 digits likely defining grant years in 8-digit numbers; (line 109) – grant year and grant supplements; (line 110) – grant year new and renewal grant applications; (line 111) – grant supplements; (line 112) – new or renewal grant applications; (line 113) – grant years.

**Line 114:** Remove the following characters in column FUTEX3: “/’, “ “, “-”, “&”.

**Lines 115 - 118:** These operations are necessary to substitute digits ‘0’, ‘1’ and ‘5’ mistyped as ‘O’, ‘I’ and ‘S’ with true digits in the DHHS grant serial numbers (right side of each number). Strings in FUTEX2 are split into the right part containing only digits and their substitute letters I, O and S, and the left part containing digit/letter mix; ‘&’s are placed as divider: (115) - Split strings containing “clean” digits on the right side; (116) – split strings containing “IOS”-mixed digit sequences on the right side; (117) – mark strings containing only letters “IOS” and digits as completely “right side substring”: (118) – remove strings that are not conforming to the above patterns.

**Line 119:** Duplicate column FUTEX2 to create FUTEX3.

**Lines 120 – 122:** Get rid of mistyped ‘0’,’1’, and ‘5’s substituted by letters ‘I’,’O’ and ‘S’: (120) – leave only left side of the grant number (any letters and digits 0-9) in column FUTEX2; (121) – leave only right part of the grant number (IOS letters and digits 0-9) in column FUTEX3; (122) – convert all ‘I’, ’S’, and ‘O’s into ‘1’, ‘5’, and ‘0’s in column FUTEX3.

**Line 123:** Concatenate column FUTEX2 and FUTEX3 to create FUTEX4.

**Lines 124 – 129:** Grant numbers in column FUTEX4 that have structure similar to DHHS grants are tagged with ‘&’ at the end, and the left part consisting of letter/digit mix (if present) is removed as follows: (124) - strings consisting of 3 to 7 digits; (125) – strings containing 2-letter institution code followed by 3 to 7 digits; (126) – strings consisting of 3-character grant activity code, 2-letter institution code, and 3 to 7 digits; (127) – strings like in previous step, but also containing leading application type digits (1, 2, 3, or 5); (128) – strings consisting of 3-letter/digit activity type and 3 to 7 digits strings like in previous step but containing leading application type digits 1, 2, 3, or 5; (129) – strings like in previous step but containing leading application type digits 1, 2, 3, or 5.

**Line 130:** Remove strings without ‘&’tag in column FUTEX4.

**Line 131:** Remove ‘&’tag in column FUTEX4.

**Line 132:** Remove characters left of the rightmost letter, including this letter in column FUTEX4.

**Line 133:** Duplicate column FUTEX4 to create FUTEX5 to compare FUTEX4 and FUTEX5 with DHHS\_GrantT3.

**Line 134:** Truncating column FUTEX4 to a substring of four character long, starting from the right.

**Lines 135 – 136:** Add one ‘0’ if strings in column FUTEX4 is shorter than 4 digits: (135) – add one ‘0’ for strings with three digits; (136) – add one ‘0’ for strings with two digits. Limited ‘0’s added to allow comparison with DHHS\_GrantT1 grant numbers while avoiding mismatches of strings with too few non- ‘0’s.

**Line 137:** Truncating column FUTEX5 to a substring of five character long, starting from the right.

**Lines 138 – 139:** Remove leading ‘0’ in column FUTEX5 in 5-digit and 4-digit strings that have at least two leading ‘0’s to prevent matching strings with just one informative digit (non-0) as it may create false positive matches.

**Line 140:** Add one leading ‘0’ to 2-digit strings in column FUTEX5 to create at least one possible match when comparing to DHHS\_GrantT1 grant numbers.

**Line 141 – 142:** Divide strings into 3-digit substrings in column FUTEX5: (141) – divide 4-digit strings into two 3-digit substrings; (142) – divide 5-digit strings into three 3-digit substrings.

**Line 143:** Place 4-digit string from column FUTEX4 and 5-digit strings in column FUTEX5 into column FUTEX5 depending on the number of grants acknowledged in column DHHS\_GrantT (more that 9, or **less/equal than 9).**

**Lines 144 - 145:** Explode substrings in column FUTEX5 into FUTEX6: (144) – convert FUTEX5 into an array column; (145) – explode FUTEX5.

**Lines 146 – 147:** Concatenate FUT column into a string column and assign codes “0”, “1”, or “2” to matches between columns FUTEX1, DHHS\_GrantT3, and FUTEX6.

**Lines 148 – 150:** Preparatory steps for concatenating columns DHHS\_GrantT1 and Match3 and grouping strings in concatenated future DHHST1tag column and Match2 column: (148) – delete numbers ‘0’ and ‘2’ (codes for unmatched strings) in column Match2; (149) – duplicate column Match2 by creating column Match3; (150) – convert ‘1’s in column Match3 into ‘%%%’ tags.

**Line 151:** Concatenate column DHHS\_GrantT1 and Match3 to create column DHHST1tag.

**Line 152 – 154:** Grouping records by PMID and FUTEX1 (152) **–** grouping rows by PMID and FUTEX1; **(153)** – concatenate column Match2 into string-type column; **(154)** – concatenate column DHHST1tag into string-type column.

**Line 155:** Reduce duplicative and multiplicative ‘1’s in column Match2 to just one “1”.

**Line 156:** In column DHHST1tag, delete all content in cells that do not contain ‘%’ characters.

**Line 157:** In column DHHST1tag, delete all grant numbers downstream of the leftmost ‘%%%’ tag.

**Line 158:** In column DHHST1tag, delete all grant numbers upstream of the one remaining ‘%%%’ tagged grant number (originally the leftmost tagged grant number) and the ‘%%%’ tag itself.

**Line 159:** Substitute grant numbers in column FUTEX1 for grant numbers In column DHHST1tag if there is code ‘1’ present in column Match2, i.e. a substring match found after comparing substrings in columns FU and DHHS\_Grant. This transformation is a necessary prerequisite to execute correct count of combined DHHS\_Grant and FU (FUTEX1) count in column ALLGrantN in step and avoid counting the same grant numbers twice.

**Line 160:** Remove ‘%’ tags in standard DHHS grant numbers.

**Lines 161 - 162:** Group by PMID and collect from column FUTEX1. Concatenate column FUTEX1 into a string-type column.

**Lines 163 – 164:** Concatenate column DHHS\_GrantT and FUTEX1 into column ALLGr.

**Lines 165 – 166:** Split column ALLGr into an array column explode it into column ALLGrant.

**Line 167:** Drop duplicative rows containing identical content in columns PMID and ALLGrant. This operation will get rid of duplicative grants between columns DHHS\_Grant and FU to obtain correct total count of grants in column ALLGrantN.

**Line 168:** Eliminate rows with “0” value in column ALLGrant. FU column and its derivative ALLGrant may contain “0” value if no Web of Science records exist for that specific PMID. Zero value is first introduced in line 5 of this module. If not removed, it adds one grant count to the total grant count in column ALLGrantN even though no grants were originally listed in column FU.

**Line 169 - 170:** Obtain cumulative count of DHHS\_Grant and FU grants by aggregating column ALLGrant and counting in column ALLGrantN.Concatenate “ALLGrant” column into string type.

**Line 171:** Adding column Pub to count with constant value of ‘1’ to be used in later transformations for counting the number of publications associated with a grant.

**Line 172 – 175:** Calculate fractional impacts for the HHS and total awards count in columns "PubPerHHSGr\_FRAC", "PubPerALLGr\_FRAC", "RCRperHHSGr\_FRAC", and "RCRperALLGr\_FRAC".

**Line 176:** Drop rows with duplicative PMID values.

**Line 177:** Select the following columns: "PMID", "PubYr", "DHHS\_GrantN", "ALLGrantN", "Pub", "RCR", "PubPerHHSGr\_FRAC", "PubPerALLGr\_FRAC", "RCRperHHSGr\_FRAC", "RCRperALLGr\_FRAC".

**Line 178:** Return dataframe.

●●●

**RJD2** (see pipeline.py lines 902 – 912)**:**

This module adds scientific impact values calculated in the RJD1 to the records of articles cited by the NCCN, ASCO, ESMO, and SITC clinical guidelines and listed in Guidelines2022 dataset.

Input dataset: Guidelines2022, RJD1

Downstream modules: AWARD\_COIN\_CALC, INV\_COIN\_CALC

|  |  |
| --- | --- |
| **Input columns:** | **Output columns:** |
| ***Guidelines2022:***  Cited\_PMID – PMIDs of the original research articles cited by clinical guidelines  Cited\_PMID\_YR – Publication year of cited article  G\_PMID\_or\_File – Guideline’s code or PMID  Guidelines\_YR – Year of publication of the guidelines  Guidelines\_Title – Guidelines title  Guidelines\_Topic – Guidelines issuer  ***RJD1:***  PMID – Article’s PubMed Identifier  PubYr – Article’s publication year  DHHS\_GrantN – Number of acknowledged RePORTER grants  Pub – publication count  RCR – Relative Citation Ratio [8]  PubPerHHSGr\_FRAC – Fractional impact of an article based on one publication count divided by the number of acknowledged RePORTER awards  PubPerALLGr\_FRAC - Fractional impact of an article based on one publication count divided by the number of all acknowledged awards  RCRperHHSGr\_FRAC - Fractional impact of an article based on article’s RCR value divided by the number of acknowledged RePORTER awards  RCRperALLGr\_FRAC – Fractional impact of an article based on article’s RCR value divided by the number of all acknowledged awards | ***RJD2:***  Cited\_PMID – PMIDs of the original research articles cited by clinical guidelines  G\_PMID\_or\_File\_LIST – List of citing guidelines codes or PMIDs  Guidelines\_YR\_LIST – List of guidelines publication years  Guidelines\_Topic\_LIST – List of guidelines issuers  SUM\_CitedPub – The number of guidelines the article was cited in  SUM\_RCRofCitedPub – RCR value multiplied by the number of guidelines the article was cited in  G\_PubPerHHSGr\_FRAC - Fractional impact of cited article based on publication count divided by the number of acknowledged RePORTER awards and multiplied by the number of citations by the guidelines  G\_PubPerALLGr\_FRAC - Fractional impact of cited article based on publication count divided by the number of all acknowledged awards and multiplied by the number of citations by the guidelines  G\_RCRperHHSGr\_FRAC - Fractional impact of cited article based on article’s RCR value divided by the number of acknowledged RePORTER awards and multiplied by the number of citations by the guidelines  G\_RCRperALLGr\_FRAC – Fractional impact of an article based on article’s RCR value divided by the number of all acknowledged awards and multiplied by the number of citations by the guidelines |

Comments:

**Line 1:** Import PySpark functions  
**Lines 2 - 3:** Define the original dataframes.

**Line 4:** Join Guidelines 2022 and RJD2 dataframes

**Line 5:** Group by PMIDs and sum publications and RCR count for each PMID. Duplicative PMIDs are added together to multiply publication impacts for articles cited in several guidelines.

**Lines 6 – 9:** Calculate fractional impact according to the formula: SUM\_CitedPub or SUM\_RCRofCitedPub divide by DHHS\_GrantN or ALLGrantN.

**Lines 10-11:** Select columns necessary for QC curation and the investigator and award COIN calculations.

●●●

**RJD3** (see pipeline.py lines 928 – 943)**:**

This module aggregates NIH RePORTER datasets that contain funding information for NIH grants funded in FY 2009-2018.

Input datasets: RePORTER\_PRJ\_FY2009, RePORTER\_PRJ\_FY2010, RePORTER\_PRJ\_FY2011, RePORTER\_PRJ\_FY2012, RePORTER\_PRJ\_FY2013, RePORTER\_PRJ\_FY2014, RePORTER\_PRJ\_FY2015, RePORTER\_PRJ\_FY2016, RePORTER\_PRJ\_FY2017, RePORTER\_PRJ\_FY2018, and RePORTER\_PRJ\_FY20019

Downstream module: AWARD\_COIN\_CALC

|  |  |
| --- | --- |
| **Input columns:** | **Output columns:** |
| ***RePORTER\_PRJ\_FY2009, RePORTER\_PRJ\_FY2010, RePORTER\_PRJ\_FY2011, RePORTER\_PRJ\_FY2012, RePORTER\_PRJ\_FY2013, RePORTER\_PRJ\_FY2014, RePORTER\_PRJ\_FY2015, RePORTER\_PRJ\_FY2016, RePORTER\_PRJ\_FY2017, RePORTER\_PRJ\_FY2018, and RePORTER\_PRJ\_FY20019:***  APPLICATION\_ID – Identification number of the award application  APPLICATION\_TYPE – The type of application (as per NIH coding\*)  CORE\_PROJECT\_NUM – Core project number  FULL\_PROJECT\_NUM – Full project number  FY – Funded fiscal year  TOTAL\_COST – Total award cost | ***RJD3:***  APPLICATION\_ID – Identification number of the award application  APPLICATION\_TYPE – The type of application (as per NIH coding\*)  CORE\_PROJECT\_NUM – Core project number  FULL\_PROJECT\_NUM – Full project number  FY – Funded fiscal year  TOTAL\_COST – Total award cost |

Comments:

**Line 1:** Import PySpark functions.

**Lines 2 – 3:** Define input dataframes.

**Lines 4 – 13:** Aggregate RePORTER\_PRJ dataframes for years 2009 – 2019.

**Line 14:** Remove activity codes from the core project numbers and delete duplicative rows.

\* <https://grants.nih.gov/grants-process/plan-to-apply/find-your-opportunity-contacts-and-due-dates/types-of-applications>

●●●

**RJD4** (see pipeline.py lines 960 – 978)**:**

This module aggregates the lists of publications PMIDs linked with supporting NIH award numbers for years 2009 through 2019. The linking datasets were downloaded from the NIH RePORTER. In addition, information about the year of publication for each article/PMID is added in column “PubYr” from RJD1.

Input datasets: RePORTER\_PUBLINK\_C\_2009, RePORTER\_PUBLINK\_C\_2010, RePORTER\_PUBLINK\_C\_2011, RePORTER\_PUBLINK\_C\_2012, RePORTER\_PUBLINK\_C\_2013, RePORTER\_PUBLINK\_C\_2014, RePORTER\_PUBLINK\_C\_2015, RePORTER\_PUBLINK\_C\_2016, RePORTER\_PUBLINK\_C\_2017, RePORTER\_PUBLINK\_C\_2018, and RePORTER\_PUBLINK\_C\_2019, and RJD1

Output module: AWARD\_COIN\_CALC

|  |  |
| --- | --- |
| **Input columns:** | **Output columns:** |
| ***RePORTER\_PUBLINK\_C\_2009, RePORTER\_PUBLINK\_C\_2010, RePORTER\_PUBLINK\_C\_2011, RePORTER\_PUBLINK\_C\_2012, RePORTER\_PUBLINK\_C\_2013, RePORTER\_PUBLINK\_C\_2014, RePORTER\_PUBLINK\_C\_2015, RePORTER\_PUBLINK\_C\_2016, RePORTER\_PUBLINK\_C\_2017, RePORTER\_PUBLINK\_C\_2018, and RePORTER\_PUBLINK\_C\_2019:***  PMID – Article’s PubMed Identifier  PROJECT\_NUMBER – Award numbers for the article-supporting awards  ***RJD1:***  PMID – Article’s PubMed Identifier  PubYr – Article’s publication year  DHHS\_GrantN – Number of acknowledged RePORTER grants  Pub – publication count  RCR – Relative Citation Ratio [8]  PubPerHHSGr\_FRAC – Fractional impact of an article based on one publication count divided by the number of acknowledged RePORTER awards  PubPerALLGr\_FRAC - Fractional impact of an article based on one publication count divided by the number of all acknowledged awards  RCRperHHSGr\_FRAC - Fractional impact of an article based on article’s RCR value divided by the number of acknowledged RePORTER awards  RCRperALLGr\_FRAC – Fractional impact of an article based on article’s RCR value divided by the number of all acknowledged awards | ***RJD4:***  PROJECT\_NUMBER – Award numbers for the article-supporting awards  PMID – Article’s PubMed Identifier  PubYr – Article’s publication year |

Comments:

**Line 1:** Import PySpark functions

**Lines 2 – 3:** Define original dataframes.

**Lines 4 – 13:** Aggregate NIH RePORTER PUBLINK dataframes for years 2009 – 2019.

**Line 14:** Abbreviate project numbers to the format compatible with other datasets.

**Line 15:** Remove duplicative rows created as a result of previous step (e.g., for R41, R42, R43, and R44 activity types).

**Lines 16 – 19:** Join with the RJD1 dataframe to add “PubYr” column defining publication years for all listed PMIDs.

●●●

**PRE\_RJD5** (see pipeline.py lines 668 – 712)**:**

This module calculates the investigator-attributed share of funding for multi-PI and/or multi-project awards. Some additional investigator categories have been created based on the analysis of the PRE\_RJD5 investigator funding data.

Input dataset: RawInvFundingByGrantAndYear

Downstream dataset: RJD5 (assembled and curated according to analyzed award categories)

|  |  |
| --- | --- |
| **Input columns:** | **Output columns:** |
| ***RawInvFundingByGrantAndYear:***  G1 – Investigator’s cohort designation  Count – Column intended for counting rows in groups  Appl\_Type – Application type  Grant\_Num – Award number  Full\_Gr\_Num – Full grant number  Sub\_Proj\_ID – Identification number of the subproject (if any)  FY – Funded fiscal year  Support\_YR – Award’s support year  Proj\_PIID – Identification number of the investigator  Proj\_PI\_Full\_Name – Full name of the investigator  Proj\_Num – In multi-project awards, project code identifying a component as a research project (p1, p2, etc), or a core (C).  Proj\_PI\_Org – Affiliation of the investigator  TC – Total cost attributed to the subproject  Title – Title of the subproject (or project in single component grants) | ***PRE\_RJD5:***  G1 – Investigator’s cohort designation  FY – Funded fiscal year  Grant\_Num – Award number  Proj\_PIID – Identification number of the investigator  Proj\_PI\_Last\_Name – Last name of the investigator  Invr\_TC\_FR – Total cost paid by the award listed in column “Grant\_Num” entry is attributed to one investigator for one year |

Comments:

**Line 1:** Import PySpark functions

**Lines 2 – 3:** Define original dataframes.

**Line 4:** Calculate the number of co-leaders in each SPORE project per grant per year. For P01s, there will be just one PI per project. These numbers will be shown in column SUM\_Count. This counting is achieved by aggregating dataframe by Proj\_Num , that shows designations “p1”, “p2”, etc. for projects and “C” for cores of multi-project grants. For single-project grants the designation in column Proj\_Num is always “p1”. This operation will provide correct count of project co-PIs in column “SUM\_Count”, but the total cost for each project will be multiplied by the number of co-PIs.  
**Lines 5 - 10:** Transform the Array columns into String columns.  
**Line 11:** Remove all rows that contain data for COREs and rename this dataframe as df1. Data for COREs will be placed into a separate dataframe in the next step. Different transformation algorithms will be performed for each dataframe.  
**Line 12:** Remove all rows that do not contain data for COREs and rename this dataframe as df2. Data for non-COREs (projects) was placed into a separate dataframe in the previous step. Each dataframe will have different transformation algorithms.  
**Line 13:** Divide TC for each row by the number of rows with identical Sub\_Proj\_IDs. This operation will re-create the original TC for each project as per step 3.  
**Line 14:** Create df3 from df1 with additional column PJperGr\_Count that will show the number of projects per grant.  
**Line 15:** Calculate the number of projects in each grant, each year separately in column SUM\_PJperGr\_Count.  
**Lines 16 - 18:** Rename columns G1, FY, and Full\_Gr\_Num to avoid having identical column names in preparation for joining df2 and df3.  
**Line 19:** Select columns "G1","FY","Proj\_Num","Full\_Gr\_Num","SUM\_PJorCORE\_TC" in df 2 preparation for joining df2 and df3.  
**Line 20:** Join the df2 & df3 to obtain fractional (per project) values for COREs total cost in later steps.  
**Line 21:** Calculate the fractional values for fractions of COREs TC attributed to each project for multi-project grants.  
**Line 22:** Select columns that are necessary for the next steps.  
**Lines 23 - 26:** Rename columns in df2 in preparation for the next step to join with df1.  
**Line 27:** Select columns in df1 that are necessary for the next steps.  
**Line 28:** Join the df1 (PJ TC each year) & df2 (combined COREs TC per PJ each year) to obtain fractional (per project) values for COREs total cost.  
**Line 29:** In preparation to creating sum of PJ\_TC and TC\_COREperPJ, substitute null values in column PplusC\_TC for “0”s. This operation is relevant for award mechanisms without cores.  
**Line 30:** Sum of PJ\_TC and TC\_COREperPJ values in new column PplusC\_TC.  
**Line 31:** Obtain TC values for each project (co)-PI funding by dividing PplusC\_TC by SUM\_Count.  
**Line 32:** Select columns in df1 that are necessary for the next steps.  
**Line 33:** Transform Proj\_PIID\_LIST column from String to Array In preparation to the next step explode transformation.  
**Line 34:** Explode Proj\_PIID\_LIST column to obtain unique record for each project co-PI in multi-PI projects/grants.  
**Line 35:** In preparation to re-adding PI last names to df1, (1) retain only columns in the original RawInvFundingByGrantAndYear df that are necessary for the merge/join step 38.  
**Line 36:** (2) Remove duplicative rows in df4. Do not use df4=df4.distinct() as it will retain duplicative PIIDs.  
**Line 37:** (3) Rename column Proj\_PIID to perform join/merge at the next step.  
**Line 38:** Join the df1 & df4 to add Project PIs names to the df1.  
**Line 39:** Remove unnecessary columns in df1.  
**Line 40:** Make all letters uppercase in column Proj\_PI\_Full\_Name.  
**Line 41:** Remove hyphens and all characters after commas in column'Proj\_PI\_Last\_Name'.  
**Line 42:** In last names (characters before commas), remove all non-letter characters in column 'Proj\_PI\_Last\_Name'.  
**Line 43:** Remove unnecessary columns in df1.  
**Line 44:** Sum all total cost for each investigator/year/grant in case an investigator has involvement in several research projects of the same multi-project grant, or there are supplements (suffixes S1, S2, etc.) to those projects.  
**Line 45:** Final outcome table

●●●

**RJD5**x**:**

This dataset contains manually curated list of investigators supported by the NCI RPG awards in FY2009-2018 and divided into several cohorts. For multi-PI and/or multi-project awards, the investigator-attributed share of funding was calculated in module PRE\_RJD5.

x *Records defining ESI or NI status of individual investigators were excluded from this dataset*

Input datasets: Assembled according to analyzed award categories

Downstream modules: INV\_COIN\_CALC, CO-ACKNOWL

|  |  |
| --- | --- |
| **Input data:** | **Output columns:** |
| This dataset was manually assembled, P01 and SPORE investigator other support (OS) group attributions were performed based on the investigator’s total cost calculations computed in module PRE\_RJD5. The data for other groups of investigators were sourced directly from PRE\_RJD5 | ***RJD5:***  G1 – Investigator’s cohort designation  FY – Funded fiscal year  Grant\_Num – Award number  Proj\_PIID – Identification number of the investigator  Proj\_PI\_Last\_Name – Last name of the investigator  Invr\_TC\_FR – Total cost paid by the award listed in column “Grant\_Num” attributed to the investigator |

●●●

**RJD6** (see pipeline.py lines 993 – 1,015)**:**

This module assembles publication data for NIH-supported investigators for years 2010 – 2019. The input data were downloaded from the NIH iCite module [7] (<https://icite.od.nih.gov/analysis>).

Input datasets: Pubs2010, Pubs2011, Pubs2012, Pubs2013, Pubs2014, Pubs2015, Pubs2016, Pubs2017, Pubs2018, and Pubs2019

Output datasets: INV\_COIN\_CALC, CO\_ACKNOWL

|  |  |
| --- | --- |
| **Input columns:** | **Output columns:** |
| ***Pubs2010, Pubs2011, Pubs2012, Pubs2013, Pubs2014, Pubs2015, Pubs2016, Pubs2017, Pubs2018, Pubs2019:***  PMID – Article’s PubMed Identifier  Pub\_Year – Article’s publication year  Author\_Last\_Name – Last names of the article’s co-authors  First\_Author – Full name of the first author  Last\_Author – Full name of the last author  Author\_Count – The number of co-authors  RCR – RCR value associated with the article  Grant\_Number – List of grants acknowledging the article | ***RJD6:***  PMID\_Inv – Article’s PubMed Identifier  Pub\_Year – Article’s publication year  Author\_Count – Number of the article’s co-authors  Gr\_Num\_Explode – One of the grant numbers acknowledging funding of the article  AuLastName\_Explode2 – Showing one of the article’s co-authors (the dataset has separate rows for each co-author with duplicative information in other columns). |

Comments:

**Line 1:** Import PySpark functions.

**lines 2 - 3:** Define the original dataframes.

**Lines 4 – 12:** Append data from the NIH RePORTER for years 2011 – 2019 into a single dataframe.

**Line 13 – 14:** Create arrays in column ‘Grant\_Number” and transform each element of the array into a separate row (explode).

**Line 15:** Truncate grant numbers in column Gr\_Num\_Explode.

**Line 16 – 17:** Create arrays in column “Author\_Last\_Name” and transform each element of the array into a separate row (explode).

**Lines 18 – 20:** To standardize spelling of complex last names, make all letters upper case and remove all non-letter characters in column AuLastName\_Explode2.

**Line 21:** In preparation for joining, remove rows with null values in column PMID\_Inv.

**Line 22:** Select columns required for the next steps.

●●●

**GRANT\_LIST**x**:**

This dataset provides a list of awards for calculation of the award-linked COINs in the module AWARD\_COIN\_CALC.

x *Records defining ESI or NI status of individual grants were excluded from this dataset*

*The investigator-specific data related to Early Stage Investigator (ESI) and New Investigator (NI) cohorts are provided only in aggregated form in datasets MEAN\_SD\_AWARD\_COIN and MEDIAN\_AWARD\_COIN.*

Input datasets: Assembled per list of awards selected for analysis

Downstream module: AWARD\_COIN\_CALC

|  |  |
| --- | --- |
| **Input columns:** | **Output columns:** |
| Assembled per list of awards selected for analysis | ***GRANT\_LIST:***  G1 – Reserved for second level grant cohort designation  G2 - Grant cohort designation  Grant – 8-character award number  Year – Award fiscal year |

●●●

**AWARD\_COIN\_CALC**x (see pipeline.py lines 11 – 59)**:**

This module calculated productivity rate (COIN) values for individual extramural awards included in the NIH RePORTER database.

Input datasets: GRANT\_LIST(x), RJD1, RJD2, RJD3, RJD4

Downstream modules: MEAN\_SD\_AWARD\_COIN, MEDIAN\_AWARD\_COIN

x *Records defining ESI or NI status of individual grants were excluded from this dataset*

*The investigator-specific data related to Early Stage Investigator (ESI) and New Investigator (NI) cohorts are provided only in aggregated form in datasets MEAN\_SD\_AWARD\_COIN and MEDIAN\_AWARD\_COIN.*

|  |  |
| --- | --- |
| **Input columns:** | **Output columns:** |
| ***GRANT\_LIST:***  G1 – Reserved for second level grant cohort designation  G2 - Grant cohort designation  Grant – 8-character award number  Year – Award fiscal year  ***RJD1:***  PMID – Article’s PubMed Identifier  PubYr – Article’s publication year  DHHS\_GrantN – Number of acknowledged RePORTER grants  Pub – publication count  RCR – Relative Citation Ratio [8]  PubPerHHSGr\_FRAC – Fractional impact of an article based on one publication count divided by the number of acknowledged RePORTER awards  PubPerALLGr\_FRAC - Fractional impact of an article based on one publication count divided by the number of all acknowledged awards  RCRperHHSGr\_FRAC - Fractional impact of an article based on article’s RCR value divided by the number of acknowledged RePORTER awards  RCRperALLGr\_FRAC – Fractional impact of an article based on article’s RCR value divided by the number of all acknowledged awards  ***RJD2:*** Cited\_PMID – PMIDs of the original research articles cited by clinical guidelines  G\_PMID\_or\_File\_LIST – List of citing guidelines codes or PMIDs  Guidelines\_YR\_LIST – List of guidelines publication years  Guidelines\_Topic\_LIST – List of guidelines issuers  SUM\_CitedPub – The number of guidelines the article was cited in  SUM\_RCRofCitedPub – RCR value multiplied by the number of guidelines the article was cited in  G\_PubPerHHSGr\_FRAC - Fractional impact of cited article based on publication count divided by the number of acknowledged RePORTER awards and multiplied by the number of citations by the guidelines  G\_PubPerALLGr\_FRAC - Fractional impact of cited article based on publication count divided by the number of all acknowledged awards and multiplied by the number of citations by the guidelines  G\_RCRperHHSGr\_FRAC - Fractional impact of cited article based on article’s RCR value divided by the number of acknowledged RePORTER awards and multiplied by the number of citations by the guidelines  G\_RCRperALLGr\_FRAC – Fractional impact of an article based on article’s RCR value divided by the number of all acknowledged awards and multiplied by the number of citations by the guidelines  ***RJD3:***  APPLICATION\_ID – Identification number of the award application  APPLICATION\_TYPE – The type of application (as per NIH coding\*)  CORE\_PROJECT\_NUM – Core project number  FULL\_PROJECT\_NUM – Full project number  FY – Funded fiscal year  TOTAL\_COST – Total award cost  ***RJD4:***  PROJECT\_NUMBER – Award numbers for the article-supporting awards  PMID – Article’s PubMed Identifier  PubYr – Article’s publication year | ***AWARD\_COIN\_CALC:***  G1 – Reserved for second level grant cohort designation  G2 - Award cohort designation  CORE\_PROJECT\_NUM – Core project number  SUMS\_TOTAL\_COST – Total cost associated with the award for the entire 2009-18 funding period  PubCostIDX\_unFR – Award productivity rate (COIN value) based on raw publication count  RCRcostIDX\_unFR – Award productivity rate (COIN value) based on raw RCR sum  PubCostIDX\_ALL – Award productivity rate (COIN value) based on fractional publication count per grant (all grants included)  RCRcostIDX\_unFR – Award productivity rate (COIN value) based on fractional RCR sum per grant (all grants included)  Gu\_PubCostIDX\_unFR – Award productivity rate (COIN value) based on raw guidelines-cited publication count  Gu\_RCRcostIDX\_unFR – Award productivity rate (COIN value) based on raw RCR sum of guidelines-cited publications  Gu\_PubCostIDX\_ALL – Award productivity rate (COIN value) based on fractional guidelines-cited publication count per grant (all grants included)  Gu\_RCRcostIDX\_unFR – Award productivity rate (COIN value) based on fractional RCR sum of guidelines-cited publications per grant (all grants included) |

Comments:

**Line 1:** Import PySpark functions  
**Lines 2 - 3:** Define original dataframes  
**Line 4:** Remove rows with duplicate values in columns "G1","G2","Grant", and "Year".  
**Line 5:** Perform joining the df with RJD3 (total cost per grants).  
**Line 6:** Sum up total cost for each grant/year.  
**Lines 7 - 9:** Transform array-type columns created after groupBy operation on line 6 ("APPL\_TYPE\_LIST", "FULL\_PJ\_NUM\_LIST", and "APPL\_ID\_LIST") into string-type columns.  
**Line 10:** Remove records for years/grants that were funded exclusively by type 3 awards. Supplemental awards are typically smaller, and may distort COIN calculations for the corresponding fiscal years.  
**Line 11:** Remove records/years associated with total funding of less than $10,000. These records may distort COIN calculations.  
**Line 12:** Create new column to match publication years (+1) for the next join step.  
**Line 13:** Join current df and RJD4 to add corresponding PMIDs and PubYr to each queried grant number. The left join is used because we want to preserve all original data on funding.  
**Line 14:** Drop column PubYr.  
**Line 15:** Rename column PMID into PMID1.  
**Line 16:** Join current df and RJD1 by PMID and publication year to calculate the overall and DHHS grants-dependent fractional scientific impact.  
**Line 17:** Select columns necessary for the next steps.  
**Lines 18 - 23:** Replace null values with "0"s in columns "Pub", "RCR", "PubPerHHSGr\_FRAC", "PubPerALLGr\_FRAC", "RCRperHHSGr\_FRAC", and "RCRperALLGr\_FRAC".  
**Line 24:** Join the current df and RJD2.  
**Lines 25 - 30:** Replace null values with "0"s in columns "SUM\_CitedPub", "SUM\_RCRofCitedPub", "G\_PubPerHHSGr\_FRAC", "G\_PubPerALLGr\_FRAC", "G\_RCRperHHSGr\_FRAC", and "G\_RCRperALLGr\_FRAC".  
**Lines 31 - 33:** Concatenate G\_PMID\_or\_File\_LIST, Guidelines\_YR\_LIST, and Guidelines\_Topic\_LIST into a string column.  
**Line 34:** Find the sums of all impact scores. Group by, collect lists, and sum columns as indicated to finish aggregation of RJD1 and RJD2 into df. For each grant/FY combination, the total cost is preserved in one cell and the impact is summed for all qualified PMIDs.  
**Line 35:** Combine all impact values calculated for single FYs in step 34 into sums of impact values calculated for all qualified FYs for each award.  
**Lines 36 - 49:** Calculate COIN values for each award from grant-specific impacts and total costs.

●●●

**MEAN\_SD\_AWARD\_COIN** (see pipeline.py lines 220 – 306)**:**

This module calculates averages (means) and standard deviations of awards' COINs.

Input datasets: AWARD\_COIN\_CALC

Downstream modules: none

|  |  |
| --- | --- |
| **Input columns:** | **Output columns:** |
| ***AWARD\_COIN\_CALC:***  G1 – Reserved for second level grant cohort designation  G2 - Grant cohort designation  CORE\_PROJECT\_NUM – Core project number  SUMS\_TOTAL\_COST – Total cost associated with the grant for the entire 2009-18 funding period  PubCostIDX\_unFR – Grant productivity rate (COIN value) based on raw publication count  RCRcostIDX\_unFR – Grant productivity rate (COIN value) based on raw RCR sum  PubCostIDX\_ALL – Grant productivity rate (COIN value) based on fractional publication count (per grant (all grants included))  RCRcostIDX\_unFR – Grant productivity rate (COIN value) based on fractional RCR sum (per grant (all grants included))  Gu\_PubCostIDX\_unFR – Grant productivity rate (COIN value) based on raw guidelines-cited publication count  Gu\_RCRcostIDX\_unFR – Grant productivity rate (COIN value) based on raw RCR sum of guidelines-cited publications  Gu\_PubCostIDX\_ALL – Grant productivity rate (COIN value) based on fractional guidelines-cited publication count (per grant (all grants included))  Gu\_RCRcostIDX\_unFR – Grant productivity rate (COIN value) based on fractional RCR sum of guidelines-cited publications (per grant (all grants included)) | ***MEAN\_SD\_AWARD\_COIN:***  G1 – Reserved for second level grant cohort designation  G2 - Grant cohort designation  N\_COUNT – Number of samples in the cohort  TC – Total cost associated with the cohort  rgRawPub\_COINav – Rounded award COIN average based on raw publication count  rgRawPub\_COINsd – Rounded award COIN standard deviation based on raw publication count  rgRawRCR\_COINav – Rounded award COIN average based on raw RCR sum  rgRawRCR\_COINsd – Rounded award COIN standard deviation based on raw RCR sum  rgALLfrPub\_COINav – Rounded award COIN average based on fractional publication count per grant (all grants)  rgALLfrPub\_COINsd – Rounded award COIN standard deviation based on fractional publication count per grant (all grants)  rgALLfrRCR\_COINav – Rounded award COIN average based on fractional RCR sum per grant (all grants)  rgALLfrRCR\_COINsd – Rounded award COIN standard deviation based on fractional RCR sum per grant (all grants)  rgGuRawPub\_COINav – Rounded award COIN average based on raw guideline-cited publication count  rgGuRawPub\_COINsd – Rounded award COIN standard deviation based on raw guideline-cited publication count  rgGuRawRCR\_COINav – Rounded award COIN average based on raw RCR sum of guideline-cited publications  rgGuRawRCR\_COINsd – Rounded award COIN standard deviation based on raw RCR sum of guideline-cited publications  rgGuALLfrPub\_COINav – Rounded award COIN average based on fractional guideline-cited publication count per grant (all grants)  rgGuALLfrPub\_COINsd – Rounded award COIN standard deviation based on fractional guideline-cited publication count per grant (all grants)  rgGuALLfrRCR\_COINav – Rounded award COIN average based on fractional RCR sum of guideline-cited publicatins per grant (all grants)  rgGuALLfrRCR\_COINsd - Rounded award COIN standard deviation based on fractional RCR sum of guideline-cited publications per grant (all grants) |

Comments:

**Line 1:** Import PySpark functions  
**Lines 2 - 3:** Define original dataframes

**Line 4:** Create column “GRANT\_COUNT” to count the number of samples in group

**Line 5:** Calculate averages and standard deviations for all groups

**Lines 6 – 87:** Calculate rounded values for all averages and standard deviations

●●●

**MEDIAN\_AWARD\_COIN** (see pipeline.py lines 404 – 530)**:**

This module calculates medians (50 pth), 75 pth, and 25 pth of the awards COINs.

Input datasets: AWARD\_COIN\_CALC

Downstream modules: none

|  |  |
| --- | --- |
| **Input columns:** | **Output columns:** |
| ***AWARD\_COIN\_CALC:***  G1 – Reserved for second level grant cohort designation  G2 - Grant cohort designation  CORE\_PROJECT\_NUM – Core project number  SUMS\_TOTAL\_COST – Total cost associated with the grant for the entire 2009-18 funding period  PubCostIDX\_unFR – Grant productivity rate (COIN value) based on raw publication count  RCRcostIDX\_unFR – Grant productivity rate (COIN value) based on raw RCR sum  PubCostIDX\_ALL – Grant productivity rate (COIN value) based on fractional publication count (per grant (all grants included))  RCRcostIDX\_unFR – Grant productivity rate (COIN value) based on fractional RCR sum (per grant (all grants included))  Gu\_PubCostIDX\_unFR – Grant productivity rate (COIN value) based on raw guidelines-cited publication count  Gu\_RCRcostIDX\_unFR – Grant productivity rate (COIN value) based on raw RCR sum of guidelines-cited publications  Gu\_PubCostIDX\_ALL – Grant productivity rate (COIN value) based on fractional guidelines-cited publication count (per grant (all grants included))  Gu\_RCRcostIDX\_unFR – Grant productivity rate (COIN value) based on fractional RCR sum of guidelines-cited publications (per grant (all grants included)) | ***MEDIAN\_AWARD\_COIN:***  G1 – Reserved for second level grant cohort designation  G2 - Grant cohort designation  N\_COUNT – Number of samples in the cohort  TC – Total cost associated with the cohort  rgRawPub\_COIN50 – Rounded award COIN median (50 pth) based on raw publication count  rgRawPub\_COIN75 – Rounded award COIN 75 pth based on raw publication count  rgRawPub\_COIN25 – Rounded award COIN 25 pth based on raw publication count  rgRawRCR\_COIN50 – Rounded award COIN median (50 pth) based on raw RCR sum  rgRawRCR\_COIN75 – Rounded award COIN 75 pth based on raw RCR sum  rgRawRCR\_COIN25 – Rounded award COIN 25 pth based on raw RCR sum  rgALLfrPub\_COIN50 – Rounded award COIN median (50 pth) based on fractional publication count  rgALLfrPub\_COIN75 – Rounded award COIN 75 pth based on fractional publication count  rgALLfrPub\_COIN25 – Rounded award COIN 25 pth based on fractional publication count  rgALLfrRCR\_COIN50 – Rounded award COIN median (50 pth) based on fractional RCR sum  rgALLfrRCR\_COIN75 – Rounded award COIN 75 pth based on fractional RCR sum  rgALLfrRCR\_COIN25 – Rounded award COIN 25 pth based on fractional RCR sum  rgGuRawPub\_COIN50 – Rounded award COIN median (50 pth) based on raw publication count of guidelines-cited publications  rgGuRawPub\_COIN75 – Rounded award COIN 75 pth based on raw publication count of guidelines-cited publications  rgGuRawPub\_COIN25 – Rounded award COIN 25 pth based on raw publication count of guidelines-cited publications  rgGuRawRCR\_COIN50 – Rounded award COIN median (50 pth) based on raw RCR sum of guidelines-cited publications  rgGuRawRCR\_COIN75 – Rounded award COIN 75 pth based on raw RCR sum of guidelines-cited publications  rgGuRawRCR\_COIN25 – Rounded award COIN 25 pth based on raw RCR sum of guidelines-cited publications  rgGuALLfrPub\_COIN50 – Rounded award COIN median (50 pth) based on fractional publication count of guidelines-cited publications  rgGuALLfrPub\_COIN75 – Rounded award COIN median 75 pth based on fractional publication count of guidelines-cited publications  rgGuALLfrPub\_COIN25 – Rounded award COIN median 25 pth based on fractional publication count of guidelines-cited publications  rgGuALLfrRCR\_COIN50 – Rounded award COIN median (50 pth) based on fractional RCR sum of guidelines-cited publications  rgGuALLfrRCR\_COIN75 – Rounded award COIN median 75 pth based on fractional RCR sum of guidelines-cited publications  rgGuALLfrRCR\_COIN25 – Rounded award COIN median 25 pth based on fractional RCR sum of guidelines-cited publications |

Comments:  
**Line 1:** Import PySpark functions  
**Lines 2-3:** define dataframes  
**Line 4:** Add new column with value "1" in each cell to count the number of investigators in each subgroup in later steps.  
**Line 5:** Calculate medians (50 pth), 75 pth, and 25 pth for each subgroup of investigators.  
**Lines 6 - 10:** Round values in column "iRawPub\_COIN50" depending on the values in each cell.  
**Lines 11 - 125:** Round values in other columns depending on the value in each cell.  
**Lines 126 - 127:** Select the resulting columns with rounded values.

●●●

**INV\_COIN\_CALC**x (see pipeline.py lines 131 – 214)**:**

This module calculates COIN values for each investigator per award mechanism.

x *Records defining ESI or NI status of individual investigators were excluded from this dataset.*

*The investigator-specific data related to Early Stage Investigator (ESI) and New Investigator (NI) cohorts are provided only in aggregated form in datasets MEAN\_SD\_INV\_COIN and MEDIAN\_INV\_COIN.*

Input datasets: RJD1, RJD2, RJD5, RJD6

Downstream modules: MEAN\_SD\_INV\_COIN, MEDIAN\_INV\_COIN

|  |  |
| --- | --- |
| **Input columns:** | **Output columns:** |
| ***RJD1:***  PMID – Article’s PubMed Identifier  PubYr – Article’s publication year  DHHS\_GrantN – Number of acknowledged RePORTER grants  Pub – publication count  RCR – Relative Citation Ratio [8]  PubPerHHSGr\_FRAC – Fractional impact of an article based on one publication count divided by the number of acknowledged RePORTER awards  PubPerALLGr\_FRAC - Fractional impact of an article based on one publication count divided by the number of all acknowledged awards  RCRperHHSGr\_FRAC - Fractional impact of an article based on article’s RCR value divided by the number of acknowledged RePORTER awards  RCRperALLGr\_FRAC – Fractional impact of an article based on article’s RCR value divided by the number of all acknowledged awards  Cited\_PMID – PMIDs of the original research articles cited by clinical guidelines  ***RJD2:***  G\_PMID\_or\_File\_LIST – List of citing guidelines codes or PMIDs  Guidelines\_YR\_LIST – List of guidelines publication years  Guidelines\_Topic\_LIST – List of guidelines issuers  SUM\_CitedPub – The number of guidelines the article was cited in  SUM\_RCRofCitedPub – RCR value multiplied by the number of guidelines the article was cited in  G\_PubPerHHSGr\_FRAC - Fractional impact of cited article based on publication count divided by the number of acknowledged RePORTER awards and multiplied by the number of citations by the guidelines  G\_PubPerALLGr\_FRAC - Fractional impact of cited article based on publication count divided by the number of all acknowledged awards and multiplied by the number of citations by the guidelines  G\_RCRperHHSGr\_FRAC - Fractional impact of cited article based on article’s RCR value divided by the number of acknowledged RePORTER awards and multiplied by the number of citations by the guidelines  G\_RCRperALLGr\_FRAC – Fractional impact of an article based on article’s RCR value divided by the number of all acknowledged awards and multiplied by the number of citations by the guidelines  ***RJD5:***  G1 – Investigator’s cohort designation  FY – Funded fiscal year  Grant\_Num – Award number  Proj\_PIID – Identification number of the investigator  Proj\_PI\_Last\_Name – Last name of the investigator  Invr\_TC\_FR – Total cost paid by the award listed in column  “Grant\_Num” attributed to one investigator, one year  ***RJD6:***  PMID\_Inv – Article’s PubMed Identifier  Pub\_Year – Article’s publication year  Author\_Count – Number of the article’s co-authors  Gr\_Num\_Explode – One of the grant numbers acknowledging funding of the article  AuLastName\_Explode2 – Showing one of the article’s co-authors (the dataset has separate rows for each co-author with duplicative information in other columns). | ***INV\_COIN\_CALC:***  G1\_ - Investigator’s cohort designation  Proj\_PIID – Identification number of the investigator  Proj\_PI\_Last\_Name – Last name of the investigator  InvestigatorTC – Total cost associated with investigator, per G1\_ group  Pub\_CC – Investigator productivity rate (COIN value) based on raw publication count  RCR\_CC – Investigator productivity rate (COIN value) based on raw RCR sum  PubPerALLGrFAuF\_CC – Investigator productivity rate (COIN value) based on fractional publication count (per grant (all grants included) and per co-author)  RCRperALLGrFAuF\_CC – Investigator productivity rate (COIN value) based on fractional RCR sum (per grant (all grants included) and per co-author)  GuCitedPub\_CC – Investigator productivity rate (COIN value) based on raw guidelines-cited publication count  GuRCRofCitedPub\_CC – Investigator productivity rate (COIN value) based on raw RCR sum of guidelines-cited publications  GuPubPerALLGrAuF\_CC – Investigator productivity rate (COIN value) based on fractional guidelines-cited publication count (per grant (all grants included) and per co-author)  GuRCRperALLGrAuF\_CC - Investigator productivity rate (COIN value) based on fractional RCR sum of guidelines-cited publications (per grant (all grants included) and per co-author) |

Comments:  
**Line 1:** import PySpark functions  
**Lines 2-3:** definitions of original dataframes  
**Lines 4-5:** Remove duplicative rows (if any) and create PubYr column by adding “1” to the FY values to match with publication years in later steps.  
**Line 6:** Truncate grant numbers to eight digits in column Grant\_Num to prepare for executing Join.  
**Line 7:** Create new column Grant\_Num\_PubYr by merging columns Grant\_Num and PubYr using dash (“-”) as delimiter. This is necessary to sum the total costs for each grant/investigator/grant mechanism combination. In the same step a list of grant numbers and corresponding publication years will be created so that it could be exploded back in the later steps without loosing connection between grant numbers and corresponding publication years.  
**Lines 8-9:** Sum Total Cost (TC) separately for each mechanism (“G1”), each investigator, and each PubYr, Division into separate years is retained to filter out years with low funding in the next step 8. The abnormally low-funded years create distorted productivity rates.  
**Line 10:** Transform array-type column Grant\_Num\_PubYr\_LIST into string-type column to prepare for the next aggregation step.  
**Line 11:** Sum TC separately for each mechanism (“G1”) and each investigator. This allows to get TC values for each investigator for all years.  
**Line 12:** Transform array-type column Grant\_Num\_PubYr\_LIST2 into string-type column to get rid of multiple strings in one cell.  
**Line 13:** Transform string-type column Grant\_Num\_PubYr\_LIST2 into a string-type column to prepare for the next Explode step.  
**Line 14:** Explode column Grant\_Num\_PubYr\_LIST2 to re-create the columns containing grant numbers and years of publication for each investigator/mechanism (G1) combination.  
**Line 15:** Perform text to columns column split of Grant\_Num\_PubYr into two columns.  
**Line 16:** Join the df and RJD6. This step adds PMIDs to each investigator/grant number/year.  
**Line 17:** Select df columns for next steps.  
**Line 18:** Select columns in RJD1 dataframe for next steps.  
**Line 19:** Join the df and RJD1. This step adds publications and their impact values to each investigator/grant number/year.  
**Lines 20-25:** For proper calculations in the next steps, substitute null values in columns "Pub", "RCR", "PubPerHHSGr\_FRAC", "PubPerALLGr\_FRAC", "RCRperHHSGr\_FRAC", and "RCRperALLGr\_FRAC" for "0"s.  
**Lines 26-31:** Calculate fractional impact for each publication based on the number of authors in each paper.  
**Line 32:** Remove duplicative PMIDs associated with the same mechanism/Proj\_PIID combinations that were created after step 19.  
**Line 33:** Sum all impact values for each mechanism(G1)/Proj\_PIID combination.  
**Line 34:** In RJD2 dataframe select columns necessary for the next steps.  
**Line 35:** Join the df and RJD2. This step adds guidelines-cited publications and their impact values to each investigator/grant number/year.  
**Lines 36-41:** For proper calculations in the next steps, substitute null values in columns "SUM\_CitedPub", "SUM\_RCRofCitedPub", "G\_PubPerHHSGr\_FRAC", "G\_PubPerALLGr\_FRAC", "G\_RCRperHHSGr\_FRAC", and "G\_RCRperALLGr\_FRAC" for "0"s.  
**Lines 42-47:** Calculate fractional impact for each publication based on the number of authors in each paper.  
**Line 48:** Remove duplicative PMIDs associated with the same mechanism/Proj\_PIID combinations that were created after step 35.  
**Line 49:** Sum all impact values for each mechanism(G1)/Proj\_PIID combination.  
**Line 50-53:** Rename columns to avoid duplicative names in the next Join steps.  
**Line 54:** Group all records by mechanism(G1)/PIID in preparation for joining dataframes with impact values.  
**Line 55:** Join the entire list of investigators and their funding support (df) with corresponding impact values in df3. The 3rd (PI Names) pair is necessary for PIs that do not have PIID and would be otherwise lumped together.  
**Line 56:** Delete columns: drop columns that are not required for the next steps.  
**Line 57:** Join the entire list of investigators and their funding support (df) with corresponding impact values in df5. The 3rd (PI Names) pair is necessary for PIs that do not have PIID and would be otherwise lumped together.  
**Line 58:** Delete columns: drop columns that are not required for the next steps.  
**Lines 59-82:** Calculate COIN values for all impact categories.  
**Line 83:** Select investigator-associated PIIDs, TC, last names, and some designated COIN categories.

●●●

**MEAN\_SD\_INV\_COIN** (see pipeline.py lines 312 – 398)**:**

This module calculated productivity rate (COIN) values for individual investigators included in the NIH RePORTER database.

Input datasets: INV\_COIN\_CALC

Downstream modules: none

|  |  |
| --- | --- |
| **Input columns:** | **Output columns:** |
| ***INV\_COIN\_CALC:***  G1\_ - Investigator’s cohort designation  Proj\_PIID – Identification number of the investigator  Proj\_PI\_Last\_Name – Last name of the investigator  InvestigatorTC – Total cost associated with investigator, per G1\_ group  Pub\_CC – Investigator productivity rate (COIN value) based on raw publication count  RCR\_CC – Investigator productivity rate (COIN value) based on raw RCR sum  PubPerALLGrFAuF\_CC – Investigator productivity rate (COIN value) based on fractional publication count (per grant (all grants included) and per co-author)  RCRperALLGrFAuF\_CC – Investigator productivity rate (COIN value) based on fractional RCR sum (per grant (all grants included) and per co-author)  GuCitedPub\_CC – Investigator productivity rate (COIN value) based on raw guidelines-cited publication count  GuRCRofCitedPub\_CC – Investigator productivity rate (COIN value) based on raw RCR sum of guidelines-cited publications  GuPubPerALLGrAuF\_CC – Investigator productivity rate (COIN value) based on fractional guidelines-cited publication count (per grant (all grants included) and per co-author)  GuRCRperALLGrAuF\_CC - Investigator productivity rate (COIN value) based on fractional RCR sum of guidelines-cited publications (per grant (all grants included) and per co-author) | ***MEAN\_SD\_INV\_COIN:***  G1\_- Grant cohort designation  N\_COUNT – Number of samples in the cohort  TC – Total cost associated with the cohort  riRawPub\_COINav – Rounded investigator COIN average based on raw publication count  riRawPub\_COINsd – Rounded investigator COIN standard deviation based on raw publication count  riRawRCR\_COINav – Rounded investigator COIN average based on raw RCR sum  riRawRCR\_COINsd – Rounded investigator COIN standard deviation based on raw RCR sum  riAuALLfrPub\_COINav – Rounded investigator COIN average based on fractional publication count per grant (all grants), per co-author  riAuALLfrPub\_COINsd – Rounded investigator COIN standard deviation based on fractional publication count per grant (all grants), per co-author  riAuALLfrRCR\_COINav – Rounded investigator COIN average based on fractional RCR sum per grant (all grants), per co-author  riAuALLfrRCR\_COINsd – Rounded investigator COIN standard deviation based on fractional RCR sum per grant (all grants), per co-author  riGuRawPub\_COINav – Rounded investigator COIN average based on raw guideline-cited publication count  riGuRawPub\_COINsd – Rounded investigator COIN standard deviation based on raw guideline-cited publication count  riGuRawRCR\_COINav – Rounded investigator COIN average based on raw RCR sum of guideline-cited publications  riGuRawRCR\_COINsd – Rounded investigator COIN standard deviation based on raw RCR sum of guideline-cited publications  riGuAuALLfrPub\_COINav – Rounded investigator COIN average based on fractional guideline-cited publication count per grant (all grants), per co-author  riGuAuALLfrPub\_COINsd – Rounded investigator COIN standard deviation based on fractional guideline-cited publication count per grant (all grants), per co-author  riGuAuALLfrRCR\_COINav – Rounded investigator COIN average based on fractional RCR sum of guideline-cited publicatins per grant (all grants), per co-author  riGuAuALLfrRCR\_COINsd – Rounded investigator COIN standard deviation based on fractional RCR sum of guideline-cited publications per grant (all grants), per co-author |

Comments:  
**Line 1:** Import PySpark functions  
**Lines 2-3:** define dataframes  
**Line 4:** Add new column with value "1" in each cell to count the number of investigators in each subgroup in later steps.  
**Line 5:** Calculate averages and standard deviations for each subgroup of investigators.  
**Lines 6 - 15:** Round values in column "iRawPub\_COINav" depending on the values in each cell.  
**Lines 16 - 85:** Round values in other columns depending on the value in each cell.  
**Lines 86 - 87:** Select the resulting columns with rounded values.

●●●

**MEDIAN\_INV\_COIN** (see pipeline.py lines 536 – 662)**:**

This module calculates and rounds medians (50 pth), 75 pth, and 25 pth of investigators COINs.

Input datasets: INV\_COIN\_CALC

Downstream modules: none

|  |  |
| --- | --- |
| **Input columns:** | **Output columns:** |
| ***INV\_COIN\_CALC:***  G1\_ - Investigator’s cohort designation  Proj\_PIID – Identification number of the investigator  Proj\_PI\_Last\_Name – Last name of the investigator  InvestigatorTC – Total cost associated with investigator, per G1\_ group  Pub\_CC – Investigator productivity rate (COIN value) based on raw publication count  RCR\_CC – Investigator productivity rate (COIN value) based on raw RCR sum  PubPerALLGrFAuF\_CC – Investigator productivity rate (COIN value) based on fractional publication count (per grant (all grants included) and per co-author)  RCRperALLGrFAuF\_CC – Investigator productivity rate (COIN value) based on fractional RCR sum (per grant (all grants included) and per co-author)  GuCitedPub\_CC – Investigator productivity rate (COIN value) based on raw guidelines-cited publication count  GuRCRofCitedPub\_CC – Investigator productivity rate (COIN value) based on raw RCR sum of guidelines-cited publications  GuPubPerALLGrAuF\_CC – Investigator productivity rate (COIN value) based on fractional guidelines-cited publication count (per grant (all grants included) and per co-author)  GuRCRperALLGrAuF\_CC - Investigator productivity rate (COIN value) based on fractional RCR sum of guidelines-cited publications (per grant (all grants included) and per co-author) | ***MEDIAN\_INV\_COIN:***  G1\_- Grant cohort designation  N\_COUNT – Number of samples in the cohort  TC – Total cost associated with the cohort  riRawPub\_COIN50 – Rounded investigator COIN median (50 pth) based on raw publication count  riRawPub\_COIN75 – Rounded investigator COIN 75 pth based on raw publication count  riRawPub\_COIN25 – Rounded investigator COIN 25 pth based on raw publication count  riRawRCR\_COIN50 – Rounded investigator COIN median (50 pth) based on raw RCR sum  riRawRCR\_COIN75 – Rounded investigator COIN 75 pth based on raw RCR sum  riRawRCR\_COIN25 – Rounded investigator COIN 25 pth based on raw RCR sum  riAuALLfrPub\_COIN50 – Rounded investigator COIN median (50 pth) based on fractional publication count per grant (all grants) per co-author  riAuALLfrPub\_COIN75 – Rounded investigator COIN 75 pth based on fractional publication count per grant (all grants) per co-author  riAuALLfrPub\_COIN25 – Rounded investigator COIN 25 pth based on fractional publication count per grant (all grants) per co-author  riAuALLfrRCR\_COIN50 – Rounded investigator COIN median (50 pth) based on fractional RCR sum per grant (all grants) per co-author  riAuALLfrRCR\_COIN75 – Rounded investigator COIN 75 pth based on fractional RCR sum per grant (all grants) per co-author  riAuALLfrRCR\_COIN25 – Rounded investigator COIN 25 pth based on fractional RCR sum per grant (all grants) per co-author  riGuRawPub\_COIN50 – Rounded investigator COIN median (50 pth) based on raw publication count of guidelines-cited publications  riGuRawPub\_COIN75 – Rounded investigator COIN 75 pth based on raw publication count of guidelines-cited publications  riGuRawPub\_COIN25 – Rounded investigator COIN 25 pth based on raw publication count of guidelines-cited publications  riGuRawRCR\_COIN50 – Rounded investigator COIN median (50 pth) based on raw RCR sum of guidelines-cited publications  riGuRawRCR\_COIN75 – Rounded investigator COIN 75 pth based on raw RCR sum of guidelines-cited publications  riGuRawRCR\_COIN25 – Rounded investigator COIN 25 pth based on raw RCR sum of guidelines-cited publications  riGuAuALLfrPub\_COIN50 – rgGuALLfrPub\_COIN50 – Rounded investigator COIN median (50 pth) based on fractional publication count of guidelines-cited publications per grant (all grants) per co-author  riGuAuALLfrPub\_COIN75 – Rounded investigator COIN median 75 pth based on fractional publication count of guidelines-cited publications per grant (all grants) per co-author  riGuAuALLfrPub\_COIN25 – Rounded investigator COIN median 25 pth based on fractional publication count of guidelines-cited publications per grant (all grants) per co-author  riGuAuALLfrRCR\_COIN50 – Rounded investigator COIN median (50 pth) based on fractional RCR sum of guidelines-cited publications per grant (all grants) per co-author  riGuAuALLfrRCR\_COIN75 – Rounded investigator COIN median 75 pth based on fractional RCR sum of guidelines-cited publications per grant (all grants) per co-author  riGuAuALLfrRCR\_COIN25 – Rounded investigator COIN median 25 pth based on fractional RCR sum of guidelines-cited publications per grant (all grants) per co-author |

Comments:  
**Line 1:** Import PySpark functions  
**Lines 2-3:** define dataframes  
**Line 4:** Add new column with value "1" in each cell to count the number of investigators in each subgroup in later steps.  
**Line 5:** Calculate medians (50 pth), 75 pth, and 25 pth for each subgroup of investigators.  
**Lines 6 - 10:** Round values in column "iRawPub\_COIN50" depending on the values in each cell.  
**Lines 11 - 125:** Round values in other columns depending on the value in each cell.  
**Lines 126 - 127:** Select the resulting columns with rounded values.

●●●

**CO\_ACKNOWL** (see pipeline.py lines 66 – 122)**:**

This module counts overlapping and unique publications of the same investigators linked to different NCI grant activities funded in the same fiscal years and calculates percentage of co-acknowledged publications for each investigator.

Note 1: When counting publications for the year of funding, change line 5 of the CO\_ACKNOWL code from df=df.withColumn('PubYr',df['FY']+1) to df=df.withColumn('PubYr',df['FY’]+0). When counting publications of the year following FY, do not change line 5. When counting publications of the year preceding the FY, change it to "-1".  
Note2: The first 16 lines of code in CO\_ACKNOWL are identical to the first 16 lines in the INV\_COIN\_CALC.

In CO\_ACKNOWL code select two investigator categories to compare in lines 4 (SAMPLE) and 7 (COMPARATOR), for example, for P01 project PIs select "P01" and "P01\_RPG&SPORE\_MATCH", for SPORE PCLs select "SPORE" and "SPORE\_RPG\_MATCH":  
line 17: df1=df.filter(df["G1"]=="P01")  
line 20: df2=df.filter(df["G1"]=="P01\_RPG&SPORE\_MATCH")

To obtain investigator-itemized count for publications linked to both SAMPLE and COMPARATOR cohorts, place return df12SCi after line 42.

To obtain investigator-itemized count for publications linked to either SAMPLE or COMPARATOR, or both cohorts, place return df12SCo after line 51.

Input datasets: RJD5, RJD6

Downstream modules: none

|  |  |
| --- | --- |
| **Input columns:** | **Output columns:** |
| ***RJD5:***  G1 – Investigator’s cohort designation  FY – Funded fiscal year  Grant\_Num – Award number  Proj\_PIID – Identification number of the investigator  Proj\_PI\_Last\_Name – Last name of the investigator  Invr\_TC\_FR – Total cost paid by the award listed in column  “Grant\_Num” attributed to one investigator, one year  ***RJD6:***  PMID\_Inv – Article’s PubMed Identifier  Pub\_Year – Article’s publication year  Author\_Count – Number of the article’s co-authors  Gr\_Num\_Explode – One of the grant numbers acknowledging funding of the article  AuLastName\_Explode2 – Showing one of the article’s co-authors (the dataset has separate rows for each co-author with duplicative information in other columns). | ***CO\_ACKNOWL:***  G1 - Grant cohort designation  G1\_1 - Grant cohort designation for COMPARATOR group  Proj\_PIID-i - Identification number of the investigators linked to co-acknowledged publications  Proj\_PI\_Last\_Name-i – Last name of the investigator linked to co-acknowledged publications  PubYr\_LIST-i – List of publication years for co-acknowledged publications  PMID\_Inv\_LIST-i – List of PMIDs of the linked co-acknowledged publications  COUNT\_Pubs-i – Counts of co-acknowledged publications per investigator  Proj\_PIID-o - Identification number of the investigators linked to all qualified publications  Proj\_PI\_Last\_Name-o – Last name of the investigator linked to all qualified publications  PubYr\_LIST-o – List of publication years for all qualified publications  PMID\_Inv\_LIST-o – List of PMIDs of the linked all qualified publications  COUNT\_Pubs-o – Counts of all qualified publications per investigator  CO\_ACKNOWLEDGED\_PERCENT – Percent of co-acknowledged publications in relation to all qualified publications |

Comments:

Line 1: import PySpark functions

Lines 2-3: definitions of original dataframes

Lines 4-5: Remove duplicative rows (if any) and create PubYr column by adding “1” to the FY values to match with publication years in later steps (see Note 1).

Line 6: Truncate grant numbers to eight digits in column Grant\_Num to prepare for executing Join.

Line 7: Create new column Grant\_Num\_PubYr by merging columns Grant\_Num and PubYr using dash (“-”) as delimiter. This is necessary to sum the total costs for each grant/investigator/grant mechanism combination. In the same step a list of grant numbers and corresponding publication years will be created so that it could be exploded back in the later steps without loosing connection between grant numbers and corresponding publication years.

Lines 8-9: Sum Total Cost (TC) separately for each mechanism (“G1”), each investigator, and each PubYr, Division into separate years is retained to filter out years with low funding in the next step 8. The abnormally low-funded years create distorted productivity rates.

Line 10: Transform array-type column Grant\_Num\_PubYr\_LIST into string-type column to prepare for the next aggregation step.

Line 11: Sum TC separately for each mechanism (“G1”) and each investigator. This allows to get TC values for each investigator for all years.

Line 12: Transform array-type column Grant\_Num\_PubYr\_LIST2 into string-type column to get rid of multiple strings in one cell.

Line 13: Transform string-type column Grant\_Num\_PubYr\_LIST2 into a string-type column to prepare for the next Explode step.

Line 14: Explode column Grant\_Num\_PubYr\_LIST2 to re-create the columns containing grant numbers and years of publication for each investigator/mechanism (G1) combination.

Line 15: Perform text to columns column split of Grant\_Num\_PubYr into two columns.

Line 16: Join the df and RJD6. This step adds PMIDs to each investigator/grant number/year.

Line 17: Select one activity from the G1 column, for example "P01" or "SPORE".

Lines 18-19: Select necessary columns from the original df. This will create duplicative rows. Remove duplicative rows.

Lines 20-23: If P01 activity was selected on line 4, select only P01-matching cohort from the original df, i.e. "P01\_RPG&SPORE\_MATCH". Select columns necessary. Rename columns using suffix “\_1” to avoid duplicative names with df1; this allows to join df1 and df2 later. Remove duplicative rows.

Line 24: Join df1 and df2 by PMIDs and investigators’ PIIDs to initiate calculation of the number of publications linked to one or both grant activity types.

Lines 25-26: Create new dataframe df\_FundYrsPub by dropping columns PMID\_Inv and PMID\_Inv\_1 from dfJ. Remove duplicative rows.

Line 27: Create a list of investigators-publications-years linked only to the SAMPLE type of grant activity (df\_FundYrsPub\_SMPL). For this purpose, retain only the first four columns in the df\_FundYrsPub dataframe.

Lines 28-29: Remove rows with null values in df\_FundYrsPub\_SMPL. Remove duplicative rows.

Line 30: Create a list of investigators-publications-years linked only to the COMPARATOR type of grant activity (df\_FundYrsPub\_COMP). For this purpose, retain only the last four columns in the df\_FundYrsPub dataframe.

Lines 31-32: Remove rows with null values in df\_FundYrsPub\_COMP. Remove duplicative rows.

Line 33: Join dataframes df\_FundYrsPub\_SMPL and df\_FundYrsPub\_COMP to find “publication years” (PubYr and PubYr\_1) that are common between the SAMPLE and COMPARATOR investigators’ cohorts (Proj\_PIID and Proj\_PIID\_1). Publication years are “funding years” plus 1 year.

Line 34: Select columns Proj\_PII and PubYr . This dataframe/list of investigators/years will be used to mark PMIDs that are common between the SAMPLE and COMPARATOR cohorts.

Line 35: Create unique names for columns Proj\_PIID and PubYr to join this dataframe with others in later steps.

Line 36: Join the df1 & dfSC to obtain the list of investigators/years/PMIDs from the SAMPLE cohort that have the same investigator/year counterparts in the COMPARATOR cohort.

Line 37: Join the df2 & dfSC to obtain the list of investigators/years/PMIDs from the COMPARATOR cohort that have the same investigator/year counterparts in the SAMPLE cohort.

Line 38: Rename columns in Proj\_PIID\_2 and PubYr\_2 into Proj\_PIID\_3 and PubYr\_3 for next step to join with df1.

Line 39: Add a new COUNTsamplePMID column into df1SC dataframe to count all SAMPLE-linked PMIDs in the next steps.

Line 40: Add a new COUNTcomparatorPMID column into df2SC dataframe to count all COMPARATOR-linked PMIDs in the next steps.

Line 41: Join df1SC and df2SC as ‘inner’ to count investigators/PMIDs pairs present in both the SAMPLE and the COMPAPATOR cohorts.

Line 42: Perform groupBy command to count publications linked to both SAMPLE and COMPARATOR cohorts for each investigator.

Line 43: Join df1SC and df2SC as ‘outer’ to count investigators/PMIDs pairs present in at least one of the SAMPLE or the COMPAPATOR cohorts.

Lines 44-48: Fill null cells in the SAMPLE-originating columns with values from the COMPARATOR-originating counterparts, i.e., Proj\_PIID, Proj\_PI\_Last\_Name, PMID\_Inv, and COUNTsamplePMID substitute for Proj\_PIID\_1, Proj\_PI\_Last\_Name\_1, PubYr\_1, and COUNTcomparatorPMID.

Lines 49-50: Remove unnecessary columns and remove rows with null values in column PMID\_Inv.

Line 51: Perform groupBy command to count publications linked to either SAMPLE or COMPARATOR cohorts, or both for each investigator. This count will only include investigators that have publications coming from at least one of the compared groups.

Lines 52-53: Rename column "Proj\_PIID" in df12SCi (for co-acknowledged publications) and in df12SC0 (for all investigator-linked publications).

Lines 54-57: Calculate percentage of co-acknowledged publications.