

Perl Regular Expressions in SDTM and ADaM Programming: Live Study Examples

Paper CT09

Pavol Valovic, Premier Research, Inc., Bratislava, Slovakia

Agenda



- ▶ Introduction to Perl Regular Expressions (PRX)
- ▶ Finding text strings in free text using PRXMATCH
- ▶ Replacing a pattern of text using PRXCHANGE
- ▶ Using both PRXMATCH and PRXCHANGE combined with other data step functions

Introduction



- ▶ Free text fields in clinical trials (AEs, CMs, protocol deviations, abnormal findings etc.)

Introduction



- ▶ Free text fields in clinical trials (AEs, CMs, protocol deviations, abnormal findings etc.)
- ▶ Need to extract information from free text fields effectively and accurately

Introduction



- ▶ Free text fields in clinical trials (AEs, CMs, protocol deviations, abnormal findings etc.)
- ▶ Need to extract information from free text fields effectively and accurately
- ▶ PRX is a useful tool for statistical programmers

Introduction



- ▶ Free text fields in clinical trials (AEs, CMs, protocol deviations, abnormal findings etc.)
- ▶ Need to extract information from free text fields effectively and accurately
- ▶ PRX is a useful tool for statistical programmers
- ▶ Scope – real-world use of PRX in clinical trial submission

Finding text strings in free text using PRXMATCH

Finding text strings in free text using PRXMATCH



- ▶ AE of Special Interest occurring in the treated (index) knee or associated tissues near the knee

Finding text strings in free text using PRXMATCH



- ▶ AE of Special Interest occurring in the treated (index) knee or associated tissues near the knee
- ▶ Anatomical location not captured in EDC

Finding text strings in free text using PRXMATCH



- ▶ AE of Special Interest occurring in the treated (index) knee or associated tissues near the knee
- ▶ Anatomical location not captured in EDC
- ▶ Conditions had to be extracted from the free text field AETERM

Finding text strings in free text using PRXMATCH



A = Event that AETERM contains the strings

("KNEE", "PATELLA", "MENISCUS", "FEMUR", "FEMORAL", "LIGAMENT", "FIBULA", "FIBULAR",
"TIBIA", "TIBIAL", "QUADRICEP TENDON", "INJECTED")

Finding text strings in free text using PRXMATCH



A = Event that AETERM contains the strings

("KNEE", "PATELLA", "MENISCUS", "FEMUR", "FEMORAL", "LIGAMENT", "FIBULA", "FIBULAR", "TIBIA", "TIBIAL", "QUADRICEP TENDON", "INJECTED")

```
if prxmatch("m/KNEE|PATELLA|MENISCUS|FEMUR|FEMORAL|LIGAMENT|FIBULA|FIBULAR|TIBIA|  
TIBIAL|QUADRICEP TENDON|INJECTED/i", aeterm) > 0  
  then condition_A = 1;  
  else condition_A = 0;
```

Finding text strings in free text using PRXMATCH



A = Event that AETERM contains the strings

("KNEE", "PATELLA", "MENISCUS", "FEMUR", "FEMORAL", "LIGAMENT", "FIBULA", "FIBULAR", "TIBIA", "TIBIAL", "QUADRICEP TENDON", "INJECTED")

```
if prxmatch("m/KNEE|PATELLA|MENISCUS|FEMUR|FEMORAL|LIGAMENT|FIBULA|FIBULAR|TIBIA|  
TIBIAL|QUADRICEP TENDON|INJECTED/i", aeterm) > 0  
then condition_A = 1;  
else condition_A = 0;
```

B = Event that AETERM contains the strings

("NON-INDEX KNEE", "NON- INDEX", "NON INDEX" "NOT INDEX KNEE", "NOT ON INDEX KNEE"
"NO INDEX KNEE", "NO INDEX", "NOT INJECT", "NON INJECT", "NOT ON INJEC", "NOT IN INJEC",
"NON-INJECT")

Finding text strings in free text using PRXMATCH



A = Event that AETERM contains the strings

("KNEE", "PATELLA", "MENISCUS", "FEMUR", "FEMORAL", "LIGAMENT", "FIBULA", "FIBULAR", "TIBIA", "TIBIAL", "QUADRICEP TENDON", "INJECTED")

```
if prxmatch("m/KNEE|PATELLA|MENISCUS|FEMUR|FEMORAL|LIGAMENT|FIBULA|FIBULAR|TIBIA|  
TIBIAL|QUADRICEP TENDON|INJECTED/i", aeterm) > 0  
then condition_A = 1;  
else condition_A = 0;
```

B = Event that AETERM contains the strings

("NON-INDEX KNEE", "NON- INDEX", "NON INDEX" "NOT INDEX KNEE", "NOT ON INDEX KNEE"
"NO INDEX KNEE", "NO INDEX", "NOT INJECT", "NON INJECT", "NOT ON INJEC", "NOT IN INJEC",
"NON-INJECT")

```
if prxmatch("m/NON-INDEX KNEE|NON- INDEX|NON INDEX|NOT INDEX KNEE|NOT ON INDEX KNEE|  
NO INDEX KNEE|NO INDEX|NOT INJECT|NON INJECT|NOT ON INJEC|  
NOT IN INJEC|NON-INJECT/i", aeterm) > 0  
then condition_B = 1;  
else condition_B = 0;
```

Replacing a pattern of text using PRXCHANGE

Replacing a pattern of text using PRXCHANGE



- ▶ Indication of concomitant medication were referenced by strings “AEnnn_” or “MHnnn_” where *nnn* is the row number of adverse event/medical history

Replacing a pattern of text using PRXCHANGE



- ▶ Indication of concomitant medication were referenced by strings “AEnnn_” or “MHnnn_” where *nnn* is the row number of adverse event/medical history
- ▶ Multiple indications were concatenated

Replacing a pattern of text using PRXCHANGE



- ▶ Indication of concomitant medication were referenced by strings “AEnnn_” or “MHnnn_” where *nnn* is the row number of adverse event/medical history
- ▶ Multiple indications were concatenated

MH003_STAGE III
CHRONIC KIDNEY
DISEASE
MH004_HYPERTENSION

Replacing a pattern of text using PRXCHANGE



- ▶ Indication of concomitant medication were referenced by strings “AEnnn_” or “MHnnn_” where *nnn* is the row number of adverse event/medical history
- ▶ Multiple indications were concatenated

MH003_STAGE III
CHRONIC KIDNEY
DISEASE
MH004_HYPERTENSION

- ▶ Remove the text strings referring to medical history records and to keep only sensible text in the listing

Replacing a pattern of text using PRXCHANGE



```
indication = prxchange('s/^[MHAЕ]{2}\d{3}_//', -1, indication);
```

Replacing a pattern of text using PRXCHANGE



```
indication = prxchange('s/^[MHAЕ]{2}\d{3}_//', -1, indication);
```

- ▶ Matches any two-letter combination of MHAЕ followed by any three digits and underscore and replaces it with blank space

Replacing a pattern of text using PRXCHANGE



```
indication = prxchange('s/^[MHAЕ]{2}\d{3}_//', -1, indication);
```

- ▶ Matches any two-letter combination of MHAЕ followed by any three digits and underscore and replaces it with blank space
- ▶ The '^' symbol matches the position at the beginning of the input string; the '-1' statement ensures that matching patterns continue to be replaced until the end of the indication string is reached

Replacing a pattern of text using PRXCHANGE



```
indication = prxchange('s/^[MHAЕ]{2}\d{3}_//', -1, indication);
```

- ▶ Matches any two-letter combination of MHAЕ followed by any three digits and underscore and replaces it with blank space
- ▶ The '^' symbol matches the position at the beginning of the input string; the '-1' statement ensures that matching patterns continue to be replaced until the end of the indication string is reached

```
indication = prxchange('s/[MHAЕ]{2}\d{3}_/+ /', -1, indication);
```

Replacing a pattern of text using PRXCHANGE



```
indication = prxchange('s/^[MHA E]{2}\d{3}_//', -1, indication);
```

- ▶ Matches any two-letter combination of MHA E followed by any three digits and underscore and replaces it with blank space
- ▶ The '^' symbol matches the position at the beginning of the input string; the '-1' statement ensures that matching patterns continue to be replaced until the end of the indication string is reached

```
indication = prxchange('s/[MHA E]{2}\d{3}_/+ /', -1, indication);
```

- ▶ Matches any two-letter combination of MHA E followed by any three digits and underscore and replaces it with '+'

Replacing a pattern of text using PRXCHANGE



MH003_STAGE III
CHRONIC KIDNEY
DISEASE
MH004_HYPERTENSION

Replacing a pattern of text using PRXCHANGE



MH003_STAGE III
CHRONIC KIDNEY
DISEASE
MH004_HYPERTENSION



```
indication = prxchange('s/^[MHA]{2}\d{3}_//', -1, indication);
```

Replacing a pattern of text using PRXCHANGE



MH003_STAGE III
CHRONIC KIDNEY
DISEASE
MH004_HYPERTENSION



```
indication = prxchange('s/^[MHA]{2}\d{3}_//', -1, indication);
```



```
indication = prxchange('s/[MHA]{2}\d{3}_/+/ ', -1, indication);
```

Replacing a pattern of text using PRXCHANGE



MH003_STAGE III
CHRONIC KIDNEY
DISEASE
MH004_HYPERTENSION



```
indication = prxchange('s/^[MHA]{2}\d{3}_//', -1, indication);
```



```
indication = prxchange('s/[MHA]{2}\d{3}_/ + /', -1, indication);
```



STAGE III CHRONIC KIDNEY
DISEASE + HYPERTENSION

Using both PRXMATCH and PRXCHANGE combined with other data step functions

Using both PRXMATCH and PRXCHANGE combined with other data step functions



- ▶ Subject ID in protocol deviation log was collected in multiple (and sometimes incorrect) ways

Using both PRXMATCH and PRXCHANGE combined with other data step functions



- ▶ Subject ID in protocol deviation log was collected in multiple (and sometimes incorrect) ways

SUBJID	PD_TEXT
	Sample Protocol Deviation text.
01-053	Sample Protocol Deviation text.
01-053	Sample Protocol Deviation text.
1104	Sample Protocol Deviation text.
1104	Sample Protocol Deviation text.
1101	Sample Protocol Deviation text.
See details in actions	Sample Protocol Deviation text.
01-122	Sample Protocol Deviation text.
1103, 1104, 1107, 1206, 1208	Sample Protocol Deviation text.
01-122	Sample Protocol Deviation text.
01-134	Sample Protocol Deviation text.
	Sample Protocol Deviation text.
01-137	Sample Protocol Deviation text.
1106	Sample Protocol Deviation text.
01-021, 01-071, 01-077	Sample Protocol Deviation text.
01-118	Sample Protocol Deviation text.
01-025	Sample Protocol Deviation text.
01-039, 01-046, 01-049, 01-070, 01-097	Sample Protocol Deviation text.
01-053, 01-055, 01-062, 01-110, 01-122, 01-128, 01-132	Sample Protocol Deviation text.
01-056	Sample Protocol Deviation text.
01-062	Sample Protocol Deviation text.
01-062	Sample Protocol Deviation text.
01-034	Sample Protocol Deviation text.

Using both PRXMATCH and PRXCHANGE combined with other data step functions



- ▶ First, we remove rows where the subject ID is either missing or not starting in the subject ID format (either correct or incorrect)

Using both PRXMATCH and PRXCHANGE combined with other data step functions



- ▶ First, we remove rows where the subject ID is either missing or not starting in the subject ID format (either correct or incorrect)

```
if missing(subjid) or (strip(subjid) ^=: '0' and strip(subjid) ^=: '1') then delete;
```

Using both PRXMATCH and PRXCHANGE combined with other data step functions



- ▶ First, we remove rows where the subject ID is either missing or not starting in the subject ID format (either correct or incorrect)

```
if missing(subjid) or (strip(subjid) ^=: '0' and strip(subjid) ^=: '1') then delete;
```

- ▶ Next, we transform incorrectly formatted subject ID ('1XXX') into correct format ('01-XXX')

Using both PRXMATCH and PRXCHANGE combined with other data step functions

- ▶ First, we remove rows where the subject ID is either missing or not starting in the subject ID format (either correct or incorrect)

```
if missing(subjid) or (strip(subjid) ^=: '0' and strip(subjid) ^=: '1') then delete;
```

- ▶ Next, we transform incorrectly formatted subject ID ('1XXX') into correct format ('01-XXX')

```
if prxmatch("m/1\d{3}/i", subjid) = 1 then subjid = prxchange("s/1(?\d{3})/01-/", -1, subjid);
```

Using both PRXMATCH and PRXCHANGE combined with other data step functions



- ▶ After those steps, all subject IDs are in correct format

Using both PRXMATCH and PRXCHANGE combined with other data step functions

- ▶ After those steps, all subject IDs are in correct format

SUBJID	PD_TEXT
01-053	Sample Protocol Deviation text.
01-053	Sample Protocol Deviation text.
1104	Sample Protocol Deviation text.
1104	Sample Protocol Deviation text.
1101	Sample Protocol Deviation text.
01-122	Sample Protocol Deviation text.
1103, 1104, 1107, 1206, 1208	Sample Protocol Deviation text.
01-122	Sample Protocol Deviation text.
01-134	Sample Protocol Deviation text.
01-137	Sample Protocol Deviation text.
1106	Sample Protocol Deviation text.
01-021, 01-071, 01-077	Sample Protocol Deviation text.
01-118	Sample Protocol Deviation text.
01-025	Sample Protocol Deviation text.
01-039, 01-046, 01-049, 01-070, 01-097	Sample Protocol Deviation text.
01-053, 01-055, 01-062, 01-110, 01-122, 01-128, 01-132	Sample Protocol Deviation text.

SUBJID	PD_TEXT
01-053	Sample Protocol Deviation text.
01-053	Sample Protocol Deviation text.
01-104	Sample Protocol Deviation text.
01-104	Sample Protocol Deviation text.
01-101	Sample Protocol Deviation text.
01-122	Sample Protocol Deviation text.
01-103, 01-104, 01-107, 01-206, 01-208	Sample Protocol Deviation text.
01-122	Sample Protocol Deviation text.
01-134	Sample Protocol Deviation text.
01-137	Sample Protocol Deviation text.
01-106	Sample Protocol Deviation text.
01-021, 01-071, 01-077	Sample Protocol Deviation text.
01-118	Sample Protocol Deviation text.
01-025	Sample Protocol Deviation text.
01-039, 01-046, 01-049, 01-070, 01-097	Sample Protocol Deviation text.
01-053, 01-055, 01-062, 01-110, 01-122, 01-128, 01-132	Sample Protocol Deviation text.

Using both PRXMATCH and PRXCHANGE combined with other data step functions



- ▶ Last step is to find the subject ID pattern and output one row per subject ID found

Using both PRXMATCH and PRXCHANGE combined with other data step functions



- ▶ Last step is to find the subject ID pattern and output one row per subject ID found
- ▶ We use PRX to determine the count of '01-XXX' patterns in the SUBJID string, which gives us the number of subject IDs

Using both PRXMATCH and PRXCHANGE combined with other data step functions

- ▶ Last step is to find the subject ID pattern and output one row per subject ID found
- ▶ We use PRX to determine the count of '01-XXX' patterns in the SUBJID string, which gives us the number of subject IDs

```
expression = prxparse("/01\-\d{3}/");  
start = 1;  
finish = length(subjid);  
count = 0;
```

Using both PRXMATCH and PRXCHANGE combined with other data step functions

- ▶ Last step is to find the subject ID pattern and output one row per subject ID found
- ▶ We use PRX to determine the count of '01-XXX' patterns in the SUBJID string, which gives us the number of subject IDs

```
expression = prxparse("/01\-\d{3}/");  
start = 1;  
finish = length(subjid);  
count = 0;
```

```
call prxnext(expression, start, finish, subjid, position, length);
```

Using both PRXMATCH and PRXCHANGE combined with other data step functions

- ▶ Last step is to find the subject ID pattern and output one row per subject ID found
- ▶ We use PRX to determine the count of '01-XXX' patterns in the SUBJID string, which gives us the number of subject IDs

```
expression = prxparse("/01\-\d{3}/");  
start = 1;  
finish = length(subjid);  
count = 0;
```

```
call prxnext(expression, start, finish, subjid, position, length);
```

```
do while (position > 0);  
  count+1;  
  call prxnext(expression, start, finish, subjid, position, length);  
end;
```

Using both PRXMATCH and PRXCHANGE combined with other data step functions



SUBJID	PD_TEXT	count
01-053	Sample Protocol Deviation text.	1
01-053	Sample Protocol Deviation text.	1
01-104	Sample Protocol Deviation text.	1
01-104	Sample Protocol Deviation text.	1
01-101	Sample Protocol Deviation text.	1
01-122	Sample Protocol Deviation text.	1
01-103, 01-104, 01-107, 01-206, 01-208	Sample Protocol Deviation text.	5
01-122	Sample Protocol Deviation text.	1
01-134	Sample Protocol Deviation text.	1
01-137	Sample Protocol Deviation text.	1
01-106	Sample Protocol Deviation text.	1
01-021, 01-071, 01-077	Sample Protocol Deviation text.	3
01-118	Sample Protocol Deviation text.	1
01-025	Sample Protocol Deviation text.	1
01-039, 01-046, 01-049, 01-070, 01-097	Sample Protocol Deviation text.	5
01-053, 01-055, 01-062, 01-110, 01-122, 01-128, 01-132	Sample Protocol Deviation text.	7
01-056	Sample Protocol Deviation text.	1
01-062	Sample Protocol Deviation text.	1
01-062	Sample Protocol Deviation text.	1
01-034	Sample Protocol Deviation text.	1
01-068	Sample Protocol Deviation text.	1
01-062, 01-069	Sample Protocol Deviation text.	2
01-070, 01-074	Sample Protocol Deviation text.	2

Using both PRXMATCH and PRXCHANGE combined with other data step functions



- ▶ A simple combination of do-loop, SCAN function, and OUTPUT statement creates one record per protocol deviation per subject

Using both PRXMATCH and PRXCHANGE combined with other data step functions



- ▶ A simple combination of do-loop, SCAN function, and OUTPUT statement creates one record per protocol deviation per subject

```
do i = 1 to count;  
  SUBJID_NEW = strip(scan(subjid, i, ','));  
  output;  
end;
```

Using both PRXMATCH and PRXCHANGE combined with other data step functions



- ▶ A simple combination of do-loop, SCAN function, and OUTPUT statement creates one record per protocol deviation per subject

```
do i = 1 to count;  
  SUBJID_NEW = strip(scan(subjid, i, ','));  
  output;  
end;
```

SUBJID	PD_TEXT	count	SUBJID_NEW
01-053	Sample Protocol Deviation text.	1	01-053
01-053	Sample Protocol Deviation text.	1	01-053
01-104	Sample Protocol Deviation text.	1	01-104
01-104	Sample Protocol Deviation text.	1	01-104
01-101	Sample Protocol Deviation text.	1	01-101
01-122	Sample Protocol Deviation text.	1	01-122
01-103, 01-104, 01-107, 01-206, 01-208	Sample Protocol Deviation text.	5	01-103
01-103, 01-104, 01-107, 01-206, 01-208	Sample Protocol Deviation text.	5	01-104
01-103, 01-104, 01-107, 01-206, 01-208	Sample Protocol Deviation text.	5	01-107
01-103, 01-104, 01-107, 01-206, 01-208	Sample Protocol Deviation text.	5	01-206
01-103, 01-104, 01-107, 01-206, 01-208	Sample Protocol Deviation text.	5	01-208
01-122	Sample Protocol Deviation text.	1	01-122
01-134	Sample Protocol Deviation text.	1	01-134
01-137	Sample Protocol Deviation text.	1	01-137
01-106	Sample Protocol Deviation text.	1	01-106
01-021, 01-071, 01-077	Sample Protocol Deviation text.	3	01-021
01-021, 01-071, 01-077	Sample Protocol Deviation text.	3	01-071
01-021, 01-071, 01-077	Sample Protocol Deviation text.	3	01-077
01-118	Sample Protocol Deviation text.	1	01-118
01-025	Sample Protocol Deviation text.	1	01-025

Conclusion



- ▶ Free text fields cause problems for statistical programmers

Conclusion



- ▶ Free text fields cause problems for statistical programmers
- ▶ A pattern matching routine using PRX in SAS effectively handles the patterns of text in the strings

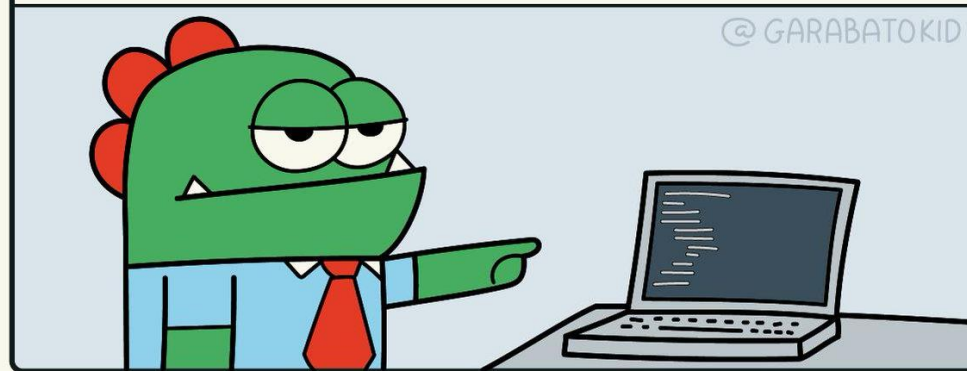
Conclusion



- ▶ Free text fields cause problems for statistical programmers
- ▶ A pattern matching routine using PRX in SAS effectively handles the patterns of text in the strings
- ▶ A good starting point for exploring the complex and beautiful world of PRX

HOW TO REGEX

STEP 1: OPEN YOUR FAVORITE EDITOR



STEP 2: LET YOUR CAT PLAY ON YOUR KEYBOARD



Contact information

Email: pavol.valovic@premier-research.com

Website: [linkedin.com/in/pavolvalovic/](https://www.linkedin.com/in/pavolvalovic/)