Introduction

A **high-frequency check (HFC)** is a check that is routinely performed on a survey/research dataset as it is being collected to monitor the quality of the data collection process and flag any potential issues. HFCs are similar in concept to the quality assurance (QA) checks that are commonly used in the tech sector for validating and cleaning server-side data; however, when referring to an HFC we make 2 important assumptions:

1. The data are collected via survey or other active collection process.
2. The intended use of the data is to answer research question(s).

These assumptions focus our definition of the "quality" of our data to more clearly mean the data's ability to provide accurate and unbiased estimates of the outcomes and covariates of interest in our research study.

At IPA, HFCs are typically implemented in Stata, after the data have been downloaded, imported, and minimally cleaned. While the types of checks included among the HFCs can vary from project to project, they typically include checks of:

- Anomalous entries or submissions (e.g. outliers, duplicates, illogical responses, etc.)
- The consistency of data across forms/survey rounds
- The functioning of the survey program
- The performance of the enumerators
- General measures of "quality" (e.g. missingness, nonresponse, timing, etc.)

Given the wealth of information they can provide, it’s hard to overstate just how important consistent implementation of HFCs are. Indeed, the ability to run faster and more detailed HFCs is one of the **MAJOR** advantages of digital data collection vis-a-vis paper.

The ipacheck Package

To help projects run HFCs more efficiently, IPA has developed the `ipacheck` Stata package. The package contains a set of user-written Stata commands that perform common checks and export the results to easy-to-read Excel documents. These

commands can roughly be divided into 4 categories: Survey Tracking, Logic Checks, Enumerator Summaries, and Research Summaries. They perform the following checks:

Survey Tracking

1. Check the progress towards productivity/recruitment goals by day and by geographic variable

Logic Checks

1. Check that all submissions are using the most recent version of the survey form
2. Check that all interviews were completed
3. Check that there are no duplicate observations
4. Check that all surveys have consent
5. Check that certain critical variables have no missing values
6. Check that follow up record information matches original
7. Check skip patterns and constraints
8. Check that no variable has all missing values
9. Check hard/soft constraints
10. Check specify other variables for items that were mismarked as 'other'
11. Check that date values fall within survey range
12. Check that there are no outliers for unconstrained variables
13. Compile all field comments
14. Check SurveyCTO text audit fields for duration per question

Enumerator Summary

1. Check the percentage of “don’t know” and “refusal” values for each variable by enumerator
2. Check the percentage giving each answer for key filter questions by enumerator
3. Check the percentage of survey refusals by enumerator
4. Check the number of surveys per day by enumerator
5. Check average interview duration by enumerator
6. Check the duration of consent and other important questions (anthropometrics, games, etc) by enumerator
7. Check the percentage of choosing "other" response by enumerator
8. Check summary statistics of key variables by enumerator

Research Summary

1. Check the frequencies of responses to key research variables.
2. Check the frequencies of responses by treatment status.
3. Check the frequencies of responses by demographic/geographic characteristics.
4. Check for any variables with low response variance.
5. Check refusal/not found rates by treatment status.

Installation

The `ipacheck.pkg` package bundles a series of user-written Stata commands created to expedite running high-frequency checks at IPA. To install, open Stata and enter one of the following:

```
/* option #1 - GitHub install (NOTE: requires internet)  */
net install ipacheck, ///
        from("https://raw.githubusercontent.com/PovertyAction/high-frequency-checks/master/ado") ///
        replace

/* option #2 - local install */
net install ipacheck, ///
        from("C:/directory/to/ado/") ///
        replace
```

Note: you will need to replace `"C:/directory/to/ado"` with the path to the ado folder in the folder.
The templates also rely on two other IPA Stata commands: `cfout` and `readreplace`. These can be installed in the Stata command window using `ssc`, i.e.

```
ssc install cfout
ssc install readreplace
```

Software Requirements

The `ipacheck` package makes extensive use of Stata's Excel modules in order to create output files that are easy to use and disseminate. Unfortunately, these features are only available as of **Stata 14.1 or later**. Therefore, users must have Stata 14.1 or greater installed on their machine prior to running the `ipacheck` templates. IPA employees with older versions of Stata should contact IT for access to a newer version.

Exercise

Below is an exercise to show you how to use the `ipacheck` package. In the exercise, we will be working with data collected from a previous IPA project. The survey instrument is a simple household survey. The data were collected using SurveyCTO, all PII have been removed and any GPS points have been anonymized.

Instructions

Section 1: Package Overview

1. To download the exercise, start by using the ipacheck command. This will initialize a folder structure, readme files, all the input sheets, and the data for the exercise:

```
ipacheck new, exercise
```

Note: if you are using the ipacheck package for your own project, you can use `ipacheck new` to download the file structure and input files without the exercise data. Use `help ipacheck` for the full functionality.

2. Now that we're in the proper directory, let's take a look at the HFC files. Start with the `04_checks/01_inputs` folder. You should see these files:
    - `hfc_inputs.xlsm` - this is the input Excel file; inside you'll find a convenient form for configuring the HFC commands.
    - `hfc_replacements.xlsm` - this is the replacement Excel file; it is a running list of edits/corrections based on HFC outputs; these replacements can be automatically added to your workflow using `readreplace`.

Now go back to the main folders, and navigate to `02_dofiles`.

- `master_check.do` - this is the master dofile; it reads the inputs, makes any replacements, runs the HFC checks, runs back checks, outputs violations and produces the dashboard.

3. Open the `master_check.do` file and review the major sections. This file is the primary controller for the HFCs, it is documented to help you understand what it's doing at each stage. All sections are set up to run according to how you set up the inputs file, buy it helps to have an idea of what is going on to be able to troubleshoot issues.

By pulling from a centralized source you'll always make sure you're starting fresh with the latest files. There are also a few other utility functions included in the `ipacheck` package including:

- `ipacheck update` - downloads the updated ado files directly from GitHub whenever IPA HQ releases an update so you don't have to go through the above Installation process again.
- `ipacheck version` - lists the current installed versions of the user-written commands (useful for verifying you have the latest installed).

Section 2: Configure the inputs

The following section will take you through the stages of setting up your HFCs. After each stage you are encouraged to run to the corresponding point in the `master_check.do` file to see the outputs created.

1. Navigate to the `04_checks/01_inputs` folder and open the `hfc_inputs.xlsm` file in Excel. This is where you'll configure the HFCs. Each logic check has its own worksheet. You'll notice, we've tried to make the input file more user-friendly by adding automatic formatting and help boxes in each sheet. In this exercise we're going to configure checks for the `survey_data.dta` dataset.
2. Open the `0. setup` sheet. This sheet is where you set global options and link to the the appropriate files necessary for running the HFCs.

- Section 1 of the `0. setup` sheet links to all the input and output files (Note: you can include file paths if the files are in separate folders). The *Master Tracking Dataset* refers to a Stata dataset containing your full sample list, either from census or previous survey waves.
- Section 2 specifies the names of the input and replacement files.
- Section 3 specifies the names of the output files.
- Section 4 specifies key variables in your survey.
- Section 5 specifies code values for don't know, missing, and not applicable. Notice, in this case, this will change all values of `-999` to `.d`, `-888` to `.r`, and `-222` to `.n`. **Keep this in mind when reading outputs and making replacements.**
- Section 6 includes options for Progress Report using the `progreport` command, which compares survey data to a master dataset for summaries of completion.
- Section 7 includes specifications for high-frequency checks.
- Section 8 specifies options for for back checks.
- Section 9 includes your SurveyCTO server and username so you can view observations from a link on the output sheet.
- Section 10 allows you to **switch on/off any check**. Even if you fill out a sheet or the specifications for a check, a check will not run if it is not turned on in Section 10. **Update the sheet with the options below.**

Data Management System

1. Datasets

Survey Dataset	../05_data/02_survey/survey_data.dta
Back Check Dataset	../05_data/03_bc/bc_survey_data.dta
Master Tracking Dataset (opt.)	../05_data/01_preloads/sample.dta
SurveyCTO Media Directory (opt.)	

2. Input Files

HFC & BC Input file	../04_checks/01_inputs/hfc_inputs.xlsm
Corrections Workbook (opt.)	../04_checks/01_inputs/hfc_replacements.xlsm
Corrections WorkSheet (opt.)	survey_data

3. Output Files

HFC Output File	../04_checks/02_outputs/hfc_outputs.xlsx
HFC Enumerator File	../04_checks/02_outputs/hfc_enumerators.xlsx
HFC Text Audit File	
Progress Report Output	../03_tracking/02_outputs/hfc_tracking.xlsx
Survey Duplicate Output File	../04_checks/02_outputs/hfc_duplicates.xlsx
Back Check Comparison Output (opt.)	../04_checks/02_outputs/bc_diffs.xlsx
HFC Research File (opt.)	../04_checks/02_outputs/hfc_research.xlsx
Replacements Log (opt.)	../04_checks/02_outputs/replacement_log.xlsx

4. Important Variables

Submission Date	submissiondate
Survey ID	id
Enumerator ID	enumid
Enumerator Team ID	supid
Back Checker ID	bcer
Back Checker Team ID	
Field Comments	
Text Audits	
Form Version	formdef_version

5. Missing Variable Codes

Missing Value (.d)	-999
Missing Value (.r)	-888
Missing Value (.n) (opt.)	-222

6. Progress Report Options

Total Number of Surveys Planned	1196
Statify Progress Report By	ward
Variables to keep in Master Data	age
Variables to keep in Survey Data (opt.)	intstatus
Save Descrepancy As (opt.)	
Target Completion Rate (opt.)	0.95
Use Variable Names as Headers (opt.)	variable
Use Values for Factors (opt.)	nolabel
ID in Master Tracking Data (opt.)	
Summary only, no individual lists (opt.)	
Export lists as separate workbooks (opt.)	

7. HFC Options

Statistics to include in Enum DB	mean sd min max
Use SD for Outliers (opt.)	sd
Use Label for Factors (opt.)	nolabel

8. BC Options

Show Unique IDs (opt.)	
Show All Discrepancies (opt.)	showall
Include All Comparisons (opt.)	
Do not Use Value labels for Factors (opt.)	nolabel
Replace Back Check Comparison File (opt.)	replace
Save Discrepancy in Stata Format	
Exclude BC Responses that Equal	
Convert All Strings to Lower	lower
Convert All Strings to Upper	
Replace Symbols with Spaces	nosymbol
Remove Leading and Trailing Blanks	trim

9. SurveyCTO Server Options

Server Name	ipahq
Username	rsandino@poverty-action.org

10. Activate Checks

Progress Report	☑	9. specify	☑
1. incomplete	☑	10. dates	☑
2. duplicates	☑	11. outliers	☑
3. consent	☑	12. field comments	☐
4. no miss	☑	13. text audits	☐
5. follow up	☑	enumdb	☑
6. logic	☑	research oneway	☑
7. all miss	☑	research twoway	☑
8. constraints	☑	backchecks	☑

3. Open the `1. incomplete` sheet and review the help boxes. This check verifies that all surveys have been completed. It corresponds with the `ipacheckcomplete` command in the `master_check.do` file. The `ipacheckcomplete` command can also check that each submission meets a minimum nonmissing entry threshold by specifying a threshold value in the `complete_percent` column. For our data set, a value of 2 in the variable `intstatus` indicates a complete interview. **Update the `variable` and `complete_value` columns to `int_status` and `2` and update the `complete_percent` column to `40` indicating that we want to flag any submission that has less than 40% of entries as nonmissing.**
4. Open the `2. duplicates` sheet and review the help boxes. This check verifies that there are no duplicate surveys. The inputs are loaded to the `ipacheckdups` command in the `master_check.do` file. **For our data set, the variables `gpsLatitude` and `gpsLongitude` should contain no duplicates. Update the variable column to reflect this.**
5. Open the `3. consent` sheet and review the help boxes. This check verifies that all surveys have consent. The inputs are loaded to the `ipacheckconsent` command in the `master_check.do` file. **For our data set, the value `1` for variable `consent` indicate consent. Update the `variable`and `consent_value` columns to reflect this.**
6. Open the `4. no miss` sheet and review the help boxes. This check verifies that certain variables have no missing values. The inputs are loaded to the `ipachecknomiss` command in the `master_check.do` file. For our dataset, the variables `gpsLatitude`, `gpsLongitude`, `enumid`, `consent`, `consentsign`, `ward`, `gender`, and `age` should have no missing values. **Update the `variable` column to reflect this.**
7. Open the `5. follow up` sheet and review the help boxes. This check verifies that respondent data at follow up matches data in the master list. The inputs are loaded to `ipacheckfollowup` in the `master_check.do` file. For our dataset, we want to verify the consistency of `gender` and `age` between the master list and the current dataset. **Add these variables to the `variable`column.**
8. Open the `6. logic` sheet and review the help boxes. This check verifies survey logic and skip patterns. The inputs are loaded to the `ipachecklogic` command. **Update the `variable`, `assert` , and `if_condition` columns with the logic checks in the table below.**

variable	label	assert	if_condition	keep
pregnant		pregnant==.	gender==0	gender
salary		salary==.	employyear==4	
occupation		occupation==.	employyear==4	
employyear		employmt==0	employyear==4	employmt

9. Open the `7. all miss` sheet and review the help boxes. This check verifies that certain variables are not all missing. The inputs are loaded to the `ipacheckallmiss` command in the `master_check.do` file. **For our data set, check all survey variables to see if any are all missing.**You can use the Stata wildcard `*` or `_all` to do it more efficiently!
10. Open the `8. constraints` sheet and review the help boxes. This check verifies hard and soft constraints. The inputs are loaded to the `ipacheckconstraints` command in the `master_check.do` file. **Update the `variable`, `soft_min`, `soft_max`, `hard_min`, and `hard_max` columns with the logic checks in the table below. Notice that you can use Stata the wildcard `*` to specify constraints for all copies of variables in a repeat group.**

variable	label	constraint	hard_min	soft_min	soft_max	hard_max
age			1	18	24	80
salary			0	0	10000	50000
childnum			0	0	5	10

11. Open the `9. specify` sheet and review the help boxes. This check lists all nonmissing specify other values to identify possible recodes or new categories. The inputs are loaded to the `ipacheckspecify` command. **Update the child and parent column with all specify other variable combinations (hint: use `ds *_other` in the command window)**
12. Open the `10. dates` sheet and review the help boxes. This check looks for common survey date errors. The inputs are loaded to the `ipacheckdates` command. **Update the `startdate`, `enddate` , and `surveystart` columns with the data in the table below.**

startdate	enddate	surveystart
starttime	endtime	11/1/2015

13. Open the `11. outliers` sheet and review the help boxes. This check looks for potential outliers in continuous variable values. The inputs are loaded to the `ipacheckoutliers` command. For our data set, we define a value 3.0 times the SD as an outlier for the variables `salary` and `childnum`. **Update the `variable` and `multiplier` columns to reflect this.**
14. Open the `enumdb` sheet and review the help boxes. This check creates the enumerator dashboard: `hfc_enumerators.xlsx`. It compiles productivity, missing, and nonresponse rates by surveyor and checks for the time spent surveying. The inputs are loaded to the `ipacheckenum`command. **Update the columns with the data in the table below.**

dkrf_variable	missing_variable	duration_variable	other_variable	stats_variable	exclude_variable	submission_date
eduattain	consent	ta_*	language_other	childnum	gender	submissiondate
occupation	consentsign		occupation_other	salary	consent	
employyear	gpsLatitude			pregnant		
childnum	gpsLongitude					
	intstatus					

15. Open the `research oneway` sheet and review the help boxes. This check creates table summaries of key research variables and outputs them to the research

file: `hfc_research.xlsx`. The type of summary (means, medians, response frequencies, etc) is determined by the variable `type` specified (e.g. continuous, categorical, binary). The inputs are loaded to the first instance of the `ipacheckresearch` command. **Update the columns with the data in the table below.**

variable	label	type
gender		cat
age		contn
edustatus		cat
eduattain		cat
employmt		cat
relationship		cat
childnum		conts

16. Open the `research twoway` sheet and review the help boxes. This check is the same as the previous but allows you to summarize key outcomes by another variable (e.g. treatment status, enumerator, region, etc.) specified in the `by` column. **Update the columns with the data in the table just as you did with research oneway, but include `treatment` in the `by' column.**
17. Open the `backchecks` sheet and review the help boxes. The columns `okrange_min`, `okrange_max`, `ttest`, and `reliability` allow for different specifications and tests, and the `type` column lets you specify what type of question each variable is. **Update the columns with the data in the table below.**

variable	label	type	okrange_min	okrange_max	ttest	reliability
gender		type 1				
age		type 1				
literacy		type 1				
language		type 1				
employmt		type 2			Yes	
occupation		type 2				Yes
salary		type 3				Yes
relationship		type 3			Yes	

Section 3: Run and Review the Output

1. Navigate to the `02_dofiles` folder to open `master_check.do` and make sure it references the correct location and input file in line 17. Run the whole do file. Once `master_check.do` has finished running, you should have an updated `hfc_outputs.xlsx` available. This file contains lists of check violations encountered by the HFC program. Open this file and inspect the contents. You'll notice it is arranged in the same format as the input with a separate sheet for each check. The output also includes a summary with overall violation counts.
2. Navigate to the `04_checks/02_outputs` folder and open the `hfc_outputs.xlsx` file. Answer the following questions:

    - **How many interviews have been conducted?**
    - **Are we missing any submissions that we planned?**
    - **Is everyone using the latest form version?**
    - **How many incomplete interviews are there? Inspect any incomplete observations and see if you can figure out what is going on. (Hint use the `list` or `browse` commands)**
    - **How many duplicates are there?**
    - **How may variables have missing values that shouldn't be missing?**
    - **How many skip pattern/logic violations are there? What can be done to prevent/resolve these?**
    - **How many constraint violations are there? Do any values appear to be nonsensical? What should be done?**
    - **Do you see any specify options that could be recoded or new categories?**
    - **Do all surveys have appropriate dates? What could be going on if not?**
3. Open the `hfc_enumerators.xlsx` file and inspect the contents. Do you notice any significant differences between the enumerators? **What should be done in response to these findings?**
4. Open the `hfc_research.xlsx` file and inspect the contents. Do the entries make sense? How would you summarize this for your PIs? **Is there anything else you might like to check?**
5. Open the `hfc_duplicates.xlsx` file and inspect the contents. Why do you think the duplicate occurred?
6. Navigate to the `03_tracking/02_outputs` folder to open the `hfc_tracking.xlsx` file and inspect the contents. How far along is survey progress? Is progress consistent by day and by ward?

Section 4: Make Replacements

1. Open the `hfc_replacements.xlsm` workbook (in the `04_checks/01_inputs` folder) and read the help boxes. This file is used to make batch corrections/edits to the survey dataset based on errors or violations found via the HFC template. You can either drop an observation, replace a value in an observation, or mark an observation as okay once you have confirmed the value and no longer want it to show up in your output files. Create a new sheet using the instruction sheet with the sheetname as `survey_data`, the ID variable as `key`, and the enumerator variable as `enumid`. When filling out this sheet, it is important to use `key` instead of the ID variable since there can be possible duplicates in your ID variable.

2. After reviewing all your output files and communicating the output of the HFCs with your field teams you discover that one of the duplicate observations is a duplicate, but has the correct values for the variables `relationship`, `pregnant`, and `childnum`. Use the replacements sheet to change `relationship` from `.d` to `1`, change `pregnant` from `.d` to `0`, change `childnum`from `.d` to `0`, and drop the duplicate with the `key` value `uuid:skdfj-aslkdjr-13lkMTDV`. Make sure to correctly specify the action as `drop`, `replace`, or `okay` for each change. **When dropping an observation, use `id` in the `variable` column and `1201` in the `value` column. This program confirms it is dropping the correct observation by ensuring the values of `variable` and `value` are correct for the key it is dropping.**
3. Add the file `hfc_replacements.xlsm` and the corresponding sheetname `survey_data` to the `0. setup` sheet of the inputs and name the file for the replacements log in Section 3. Verify that the replacements were made and the errors no longer appear in the output file. Inspect the output file for other potential replacements that can be made and add them to the list.

Section 5: Rerun the HFCs

1. Rerun the `master_check.do` and verify that the replacements were made and the errors no longer appear in the output file. Inspect the output file for other potential replacements that can be made and add them to the list.