# Summary report for UK introductions

————————————————————————————TypeError Traceback (most recent call last) in 1 current_date = time.make_current_week(week) 2 —-> 3 intro_bigs, intro_smalls, intro_alls, count, intro_countries, intro_object_dict, omitted, taxa, new_lineages, taxon_dictionary, most_recent_sample, intro_int_list, unclear_taxa = parse.make_objects(current_date, metadata_file) 4 singletons_count, smalls_count = descrip.get_preliminary_info(intro_countries) 5 ~/anaconda3/envs/report/lib/python3.7/site-packages/UK_full_report/utils/data_parsing.py in make_objects(current_day, metadata_file) 94 for intro, taxa in intros_to_taxa.items(): 95 —> 96 i_o = case_def.introduction(intro, taxa, most_recent_sample, current_week) 97 #i_o.overall_lineage = taxa[0].overall_lineage 98 i_o.acctrans_designations = intro_acctrans[i_o.id] ~/anaconda3/envs/report/lib/python3.7/site-packages/UK_full_report/utils/case_definitions.py in **init**(self, name, taxa, current_day, current_week) 84 self.get_global_lineages() 85 self.get_date_loc_info(current_day) —> 86 self.define_status(current_week) 87 88 ~/anaconda3/envs/report/lib/python3.7/site-packages/UK_full_report/utils/c in define_status(self, current_week) 131 int_list = [] 132 for k,v in self.epiweek_counts.items(): –> 133 if k in last_month: 134 int_list.append(v) 135 ~/anaconda3/envs/report/lib/python3.7/site-packages/epiweeks.py in **eq**(self, other) 45 46 def **eq**(self, other: “Week”) -> bool: —> 47 return self._compare(other) == 0 48 49 def **gt**(self, other: “Week”) -> bool: ~/anaconda3/envs/report/lib/python3.7/site-packages/epiweeks.py in _compare(self, other) 64 other_name = type(other).**name** 65 if not isinstance(other, self.**class**): —> 66 raise TypeError(f“can’t compare ‘{class_name}’ to ‘{other_name}’ ”) 67 if self._system != other._system: 68 raise TypeError( TypeError: can’t compare ‘Week’ to ‘str’

Note: the size of a lineage may be due to a low amount of transmission of this lineage, but it is likely also that it just hasn’t been sampled as frequently, especially for newer lineages.

Sequences which were replicates or too error-prone were removed from this analysis.

————————————————————————————NameError Traceback (most recent call last) in —> 1 print(str(smalls_count) + ” are introductions which only contained five sequences or fewer, and so have been left out of visualisation in the interests of clarity“) NameError: name ‘smalls_count’ is not defined

————————————————————————————NameError Traceback (most recent call last) in 1 #split = lin_exp.find_splits(intro_alls) 2 —-> 3 status_counts, reactivated_lineages, continuining_lineages = lin_exp.describe_lineages(intro_bigs) 4 5 reactivateds = status_counts[“Reactivated”] NameError: name ‘intro_bigs’ is not defined

The following table contains information about lineages and the number of sequences in each country in the UK for each lineage, in reverse size order. Each entry is the count of sequences from each lineage in each country, with the percentage of the total sequences from that lineage that this count represents.

It is also written to “summary_files” as “introduction_summary.tsv” for further use.

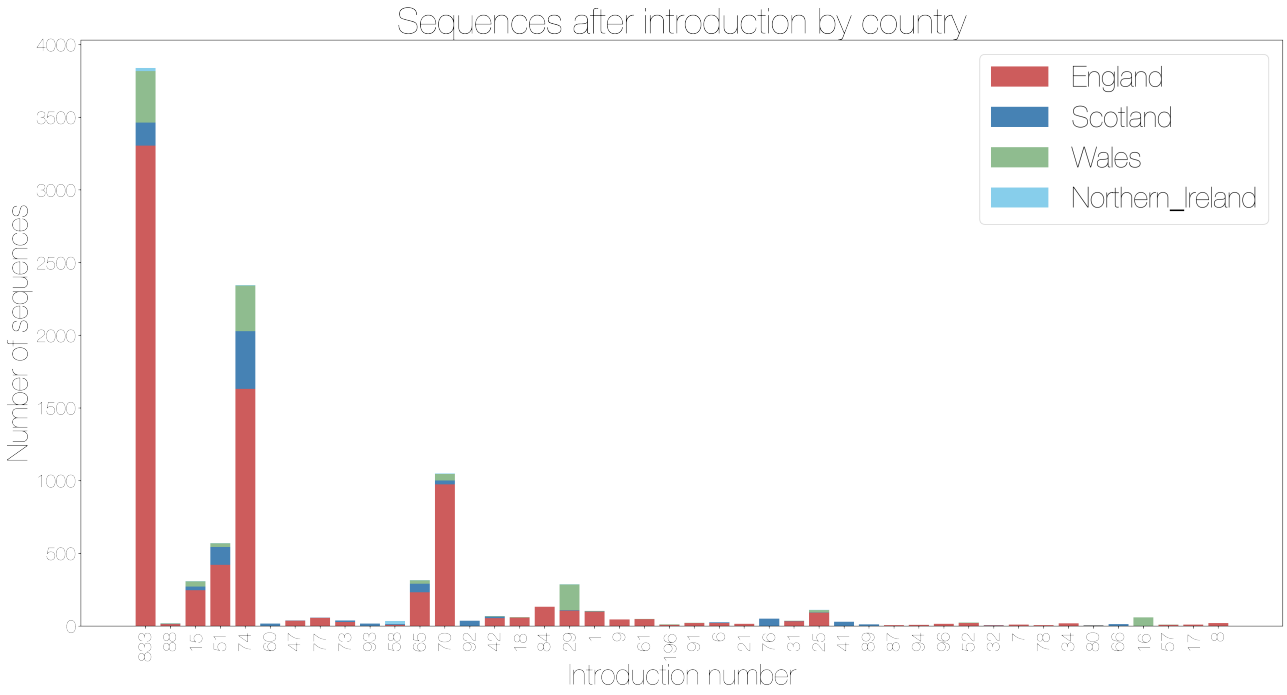
————————————————————————————NameError Traceback (most recent call last) in —> 1 intro_country_counts, intro_country_percentages, intro_country_together, intro_total_numbers = descrip.prep_dicts(intro_countries) 2 dataframe, tree_order = descrip.make_dataframe(intro_country_together, intro_total_numbers, intro_object_dict) 3 4 print(dataframe.to_markdown()) NameError: name ‘intro_countries’ is not defined

---------------------------------------------------------------------------NameError
Traceback (most recent call last)<ipython-input-1-5da465605b14> in
<module>
----> 1 writing.write_summary_files(output_directory, dataframe,
omitted, week)
NameError: name 'dataframe' is not defined

These data is represented in the stacked bar chart below. Note that the number of sequences is likely to

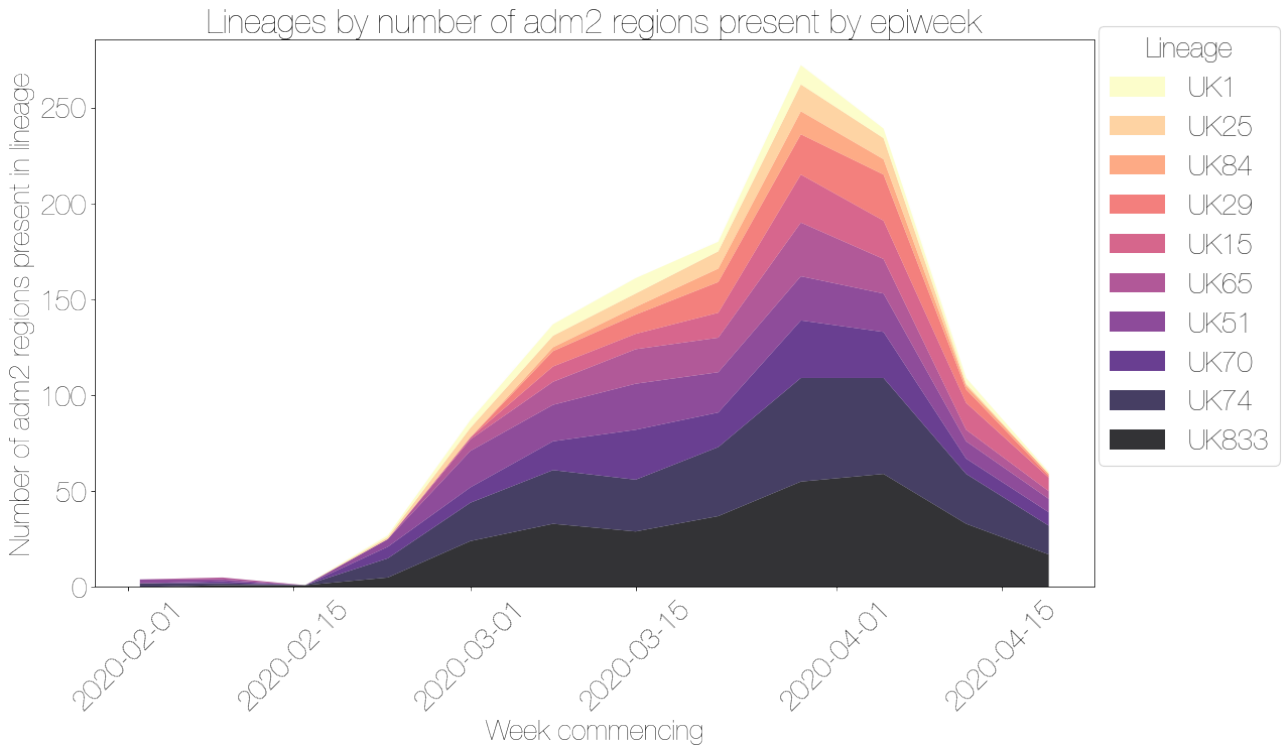
be due more to differing sampling efforts in different regions, rather than genuine differences in numbers of cases.

---------------------------------------------------------------------------NameError
Traceback (most recent call last)<ipython-input-1-82848448dadb> in
<module>
----> 1 df_counts, df_thinned, df_acctrans_counts =
dp.make_plotting_dfs(intro_country_counts, intro_object_dict)
2
3 dp.plot_bars(intro_bigs)
NameError: name 'intro_country_counts' is not defined



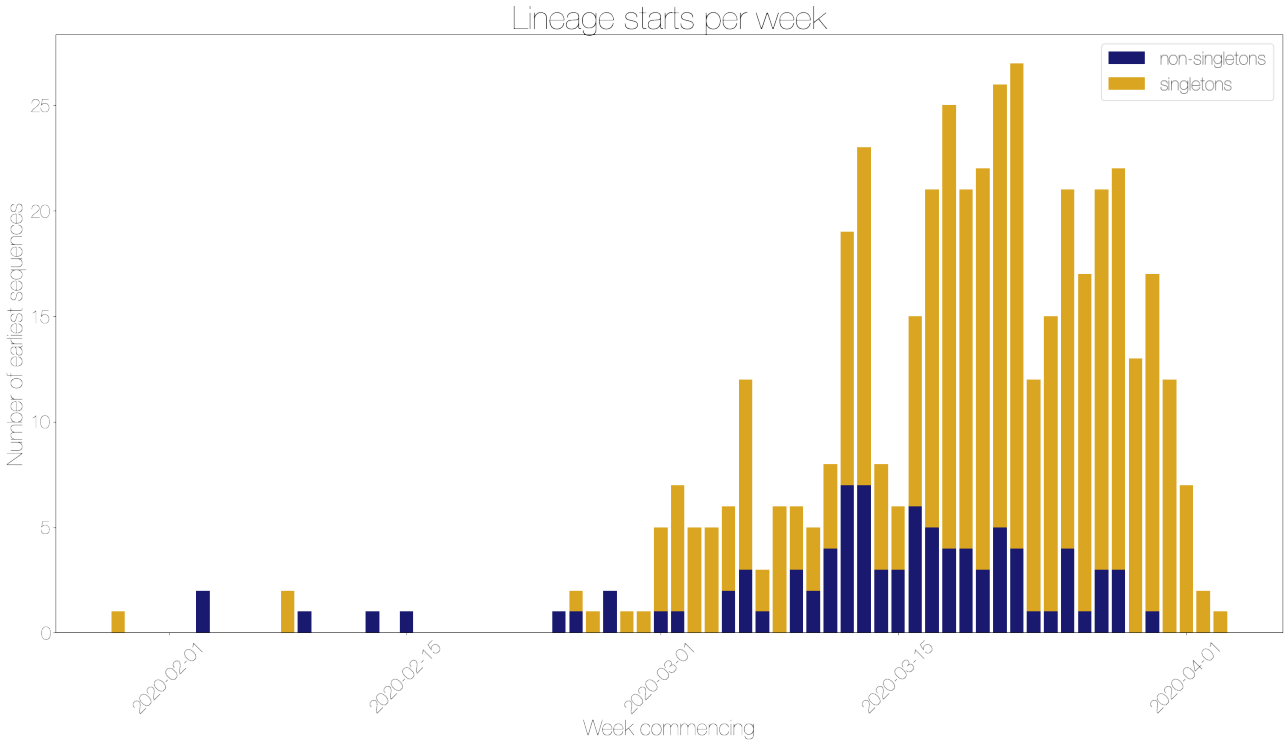

The relative growth and decline of the ten most sampled lineages in terms of number of counties they are present in is shown below.

---------------------------------------------------------------------------NameError
Traceback (most recent call last)<ipython-input-1-07194fec9b74> in
<module>
----> 1 dp.plot_geog_plot(intro_bigs, False)
NameError: name 'intro_bigs' is not defined



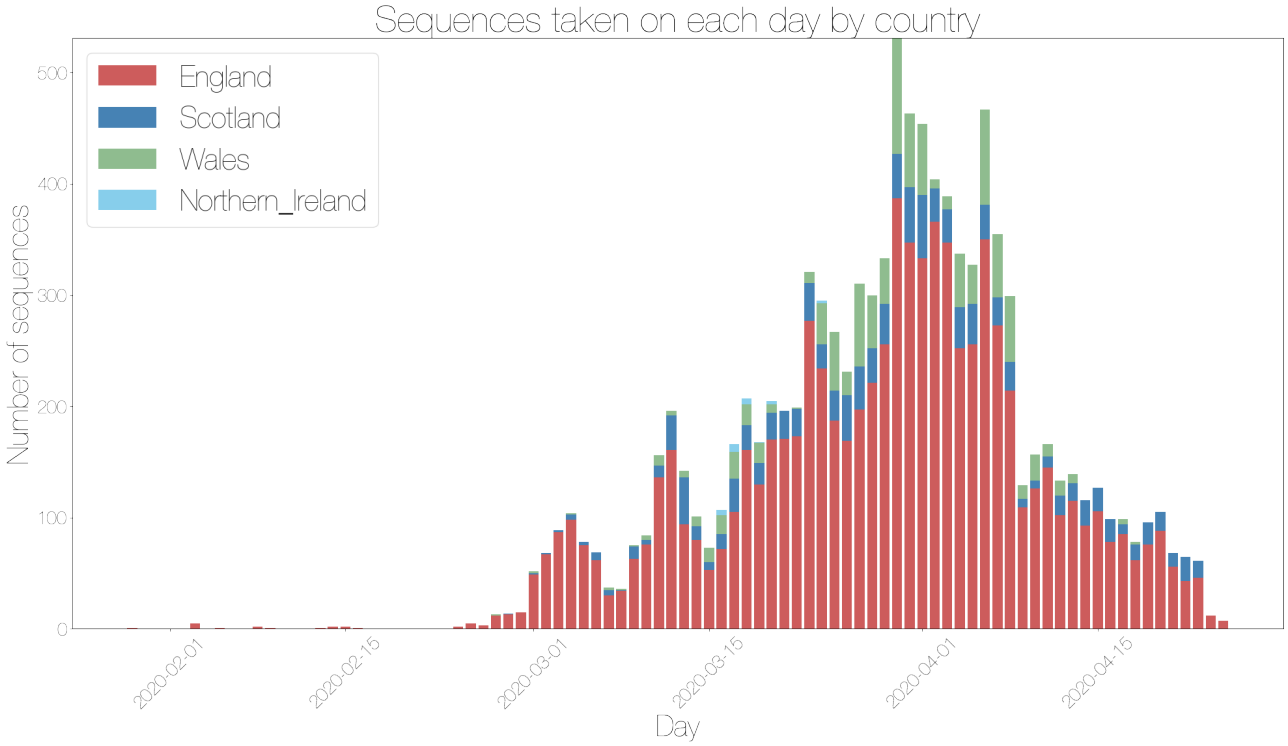

The date of first sequence in the cluster is shown below for every cluster with date information.

---------------------------------------------------------------------------NameError
Traceback (most recent call last)<ipython-input-1-ca69f79d8c00> in
<module>
----> 1 dp.plot_starts(intro_alls)
NameError: name 'intro_alls' is not defined



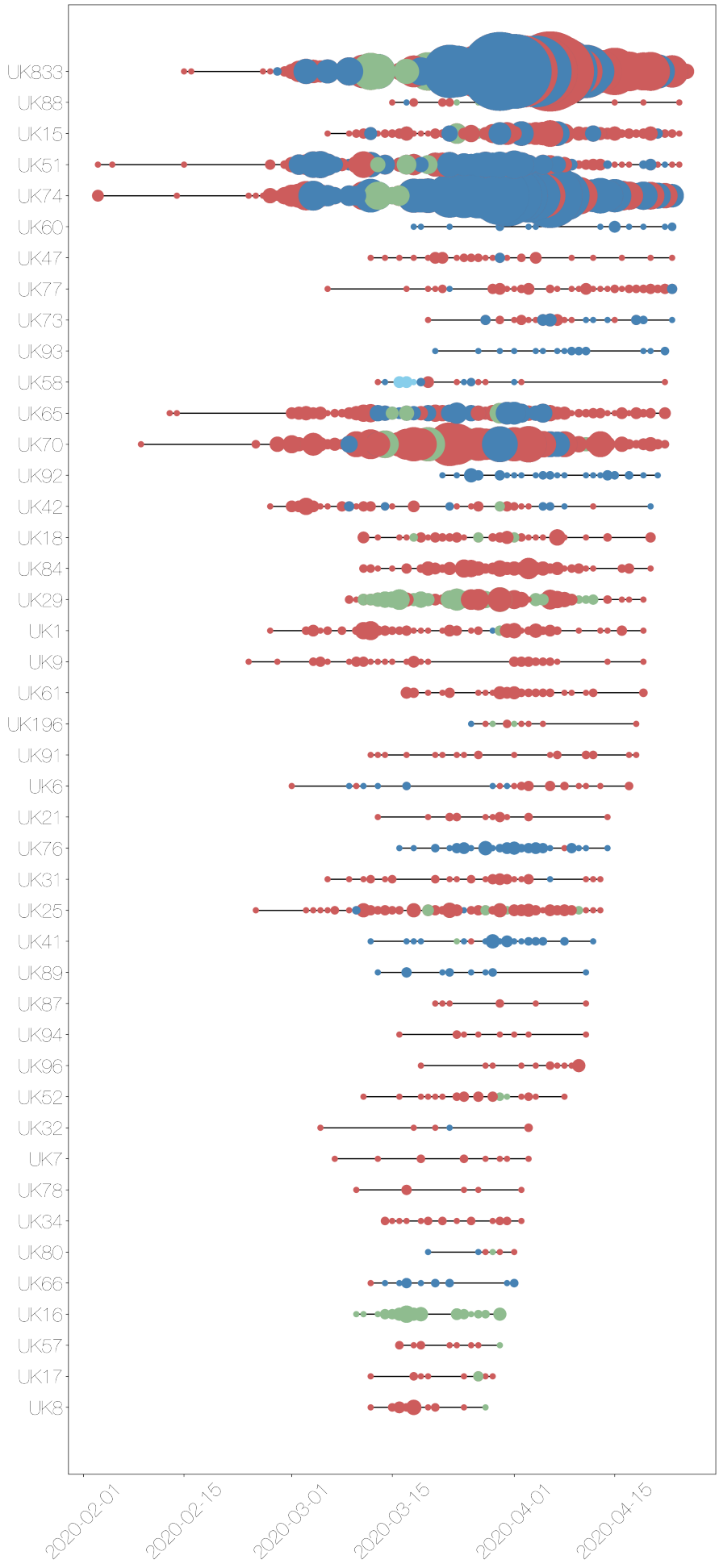

For comparison, here is a plot of the day that every sequence was taken, coloured by country. Note that sequences without dates were not included.

---------------------------------------------------------------------------NameError
Traceback (most recent call last)<ipython-input-1-5de6eef29455> in
<module>
----> 1 dp.plot_sequences_over_time(taxa)
NameError: name 'taxa' is not defined




These introductions are shown on the timeline below. Each line represents the length of the cluster, from oldest to most recent sampling date. The dots are sized by the number of sequences taken on that date, and again are colour coded by country.

———————————————————————NameError Traceback (most recent call last) in —-> 1 dp.make_timeline(intro_bigs) NameError: name ‘intro_bigs’ is not defined

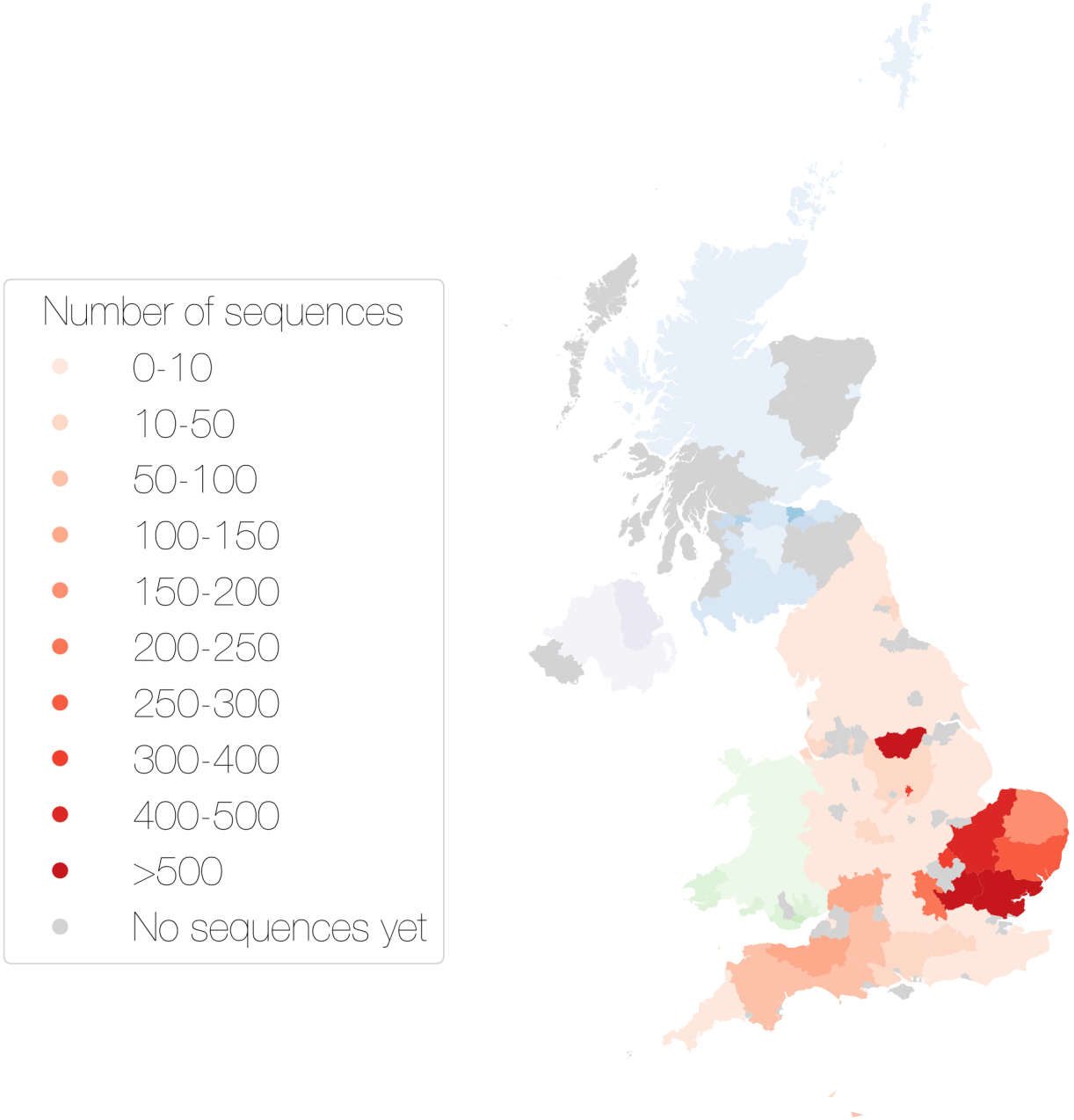

UK833
UK88
UK15
UK51
UK74
UK60
UK47
UK77
UK73
UK93
UK58
UK65
UK70
UK92
UK42
UK18
UK84
UK29
UK1
UK9
UK61
UK196
UK91
UK6
UK21
UK76
UK31
UK25
UK41
UK89
UK87
UK94
UK96
UK52
UK32
UK7
UK78
UK34
UK80
UK66
UK16
UK57
UK17
UK8
2020-02-01
2020-02-15
2020-03-01
2020-03-15
2020-04-01
2020-04-15

Error in callback (for post_execute):

———————————————————————————KeyboardInterrupt Traceback (most recent call last)~/anaconda3/envs/report/lib/python3.7/site-packages/ipykernel/pylab/backend_inline.py in flush_figures() 119 # ignore the tracking, just draw and close all figures 120 try: –> 121 return show(True) 122 except Exception as e: 123 # safely show traceback if in IPython, else raise ~/anaconda3/envs/report/lib/python3.7/site-packages/ipykernel/pylab/backend_inline.py in show(close, block) 41 display( 42 figure_manager.canvas.figure, —> 43 metadata=_fetch_figure_metadata(figure_manager.canvas.figure) 44 ) 45 finally: ~/anaconda3/envs/report/lib/python3.7/site-packages/IPython/core/display.py in display(include, exclude, metadata, transient, display_id, *objs,* ***kwargs)*** ***311 publish_display_data(data=obj, metadata=metadata,*** *kwargs) 312 else: –> 313 format_dict, md_dict = format(obj, include=include, exclude=exclude) 314 if not format_dict: 315 # nothing to display (e.g.* ipython_display *took over) ~/anaconda3/envs/report/lib/python3.7/site-packages/IPython/core/formatters.py in format(self, obj, include, exclude) 178 md = None 179 try: –> 180 data = formatter(obj) 181 except: 182 # FIXME: log the exception in* ***call****(self, obj) ~/anaconda3/envs/report/lib/python3.7/site-packages/IPython/core/formatters.py in catch_format_error(method, self,* args, **kwargs) 222 “”“show traceback on failed format call”“” 223 try: –> 224 r = method(self, *args, **kwargs) 225 except NotImplementedError: 226 # don’t warn on NotImplementedErrors ~/anaconda3/envs/report/lib/python3.7/site-packages/IPython/core/formatters.py in call(self, obj) 339 pass 340 else: –> 341 return printer(obj) 342 # Finally look for special method names 343 method = get_real_method(obj, self.print_method) ~/anaconda3/envs/report/lib/python3.7/site-packages/IPython/core/pylabtools.py in (fig) 252 jpg_formatter.for_type(Figure, lambda fig: print_figure(fig, ‘jpg’,* kwargs)) 253 if ‘svg’ in formats: –> 254 svg_formatter.for_type(Figure, lambda fig: print_figure(fig, ‘svg’, **kwargs)) 255 if ‘pdf’ in formats: 256 pdf_formatter.for_type(Figure, lambda fig: print_figure(fig, ‘pdf’,* kwargs)) ~/anaconda3/envs/report/lib/python3.7/site-packages/IPython/core/pylabtools.py in print_figure(fig, fmt, bbox_inches, **kwargs) 130 FigureCanvasBase(fig) 131 –> 132 fig.canvas.print_figure(bytes_io,* kw) 133 data = bytes_io.getvalue() 134 if fmt == ‘svg’: ~/anaconda3/envs/report/lib/python3.7/site-packages/matplotlib/backend_bases.py in print_figure(self, filename, dpi, facecolor, edgecolor, orientation, format, bbox_inches, **kwargs) 2077 print_method, dpi=dpi, orientation=orientation), 2078 draw_disabled=True) -> 2079 self.figure.draw(renderer) 2080 bbox_artists = kwargs.pop(“bbox_extra_artists”, None) 2081 bbox_inches = self.figure.get_tightbbox(renderer, ~/anaconda3/envs/report/site-packages/matplotlib/artist.py in draw_wrapper(artist, renderer, *args, **kwargs) 36 renderer.start_filter() 37 —> 38 return draw(artist, renderer, *args, **kwargs) 39 finally: 40 if artist.get_agg_filter() is not None: ~/anaconda3/envs/report/lib/python3.7/site-packages/matplotlib/figure.py in draw(self, renderer) 1734 self.patch.draw(renderer) 1735 mimage._draw_list_compositing_images( -> 1736 renderer, self, artists, self.suppressComposite) 1737 1738 renderer.close_group(‘figure’) ~/anaconda3/envs/report/lib/python3.7/site-packages/matplotlib/image.py in _draw_list_compositing_images(renderer, parent, artists, suppress_composite) 135 if not_composite or not has_images: 136 for a in artists: –> 137 a.draw(renderer) 138 else: 139 # Composite any adjacent images together ~/anaconda3/envs/report/lib/python3.7/site-packages/matplotlib/artist.py in draw_wrapper(artist, renderer, *args, **kwargs) 36 renderer.start_filter() 37 —> 38 return draw(artist, renderer, *args, **kwargs) 39 finally: 40 if artist.get_agg_filter() is not None: ~/anaconda3/envs/report/lib/python3.7/site-packages/matplotlib/axes/_base.py in draw(self, renderer, inframe) 2628 renderer.stop_rasterizing() 2629 -> 2630 mimage._draw_list_compositing_images(renderer, self, artists) 2631 2632 renderer.close_group(‘axes’) ~/anaconda3/envs/report/lib/python3.7/site-packages/matplotlib/image.py in _draw_list_compositing_images(renderer, parent, artists, suppress_composite) 135 if not_composite or not has_images: 136 for a in artists: –> 137 a.draw(renderer) 138 else: 139 # Composite any adjacent images together ~/anaconda3/envs/report/lib/python3.7/site-packages/matplotlib/artist.py in draw_wrapper(artist, renderer, *args, **kwargs) 36 renderer.start_filter() 37 —> 38 return draw(artist, renderer, *args, **kwargs) 39 finally: 40 if artist.get_agg_filter() is not None: ~/anaconda3/envs/report/lib/python3.7/site-packages/matplotlib/axis.py in draw(self, renderer, *args, **kwargs) 1237 # the actual bbox 1238 -> 1239 self._update_label_position(renderer) 1240 1241 self.label.draw(renderer) ~/anaconda3/envs/report/lib/python3.7/site-packages/matplotlib/axis.py in _up-

date_label_position(self, renderer) 2322 # use axes if spine doesn't exist 2323 spinebbox = self.axes.bbox -> 2324 bbox = mtransforms.Bbox.union(bboxes + [spinebbox]) 2325 left = bbox.x0 2326 self.label.set_position( ~/anaconda3/envs/report/lib/python3.7/site-packages/matplotlib/transforms.py in union(bboxes) 701 # can remove once we are at numpy >= 1.15 702 with np.errstate(invalid='ignore'): –> 703 x0 = np.min([bbox.xmin for bbox in bboxes]) 704 x1 = np.max([bbox.xmax for bbox in bboxes]) 705 y0 = np.min([bbox.ymin for bbox in bboxes]) ~/anaconda3/envs/report/lib/python3.7/site-packages/matplotlib/transforms.py in (.0) 701 # can remove once we are at numpy >= 1.15 702 with np.errstate(invalid='ignore'): –> 703 x0 = np.min([bbox.xmin for bbox in bboxes]) 704 x1 = np.max([bbox.xmax for bbox in bboxes]) 705 y0 = np.min([bbox.ymin for bbox in bboxes]) ~/anaconda3/envs/report/lib/python3.7/site-packages/matplotlib/transforms.py in xmin(self) 347 def xmin(self): 348 ""“The left edge of the bounding box.”"" –> 349 return np.min(self.get_points()[:, 0]) 350 351 @property <**array_function** internals> in amin(*args, **kwargs) ~/anaconda3/envs/report/lib/python3.7/site-packages/numpy/core/fromnumeric.py in amin(a, axis, out, keepdims, initial, where) 2791 “”" 2792 return _wrapreduction(a, np.minimum, 'min', axis, None, out, -> 2793 keepdims=keepdims, initial=initial, where=where) 2794 2795 ~/anaconda3/envs/report/lib/python3.7/site-packages/numpy/core/fromnumeric.py in _wrapreduction(obj, ufunc, method, axis, dtype, out, **kwargs) 88 return reduction(axis=axis, out=out, passkwargs) 89 —> 90 return ufunc.reduce(obj, axis, dtype, out, **passkwargs) 91 92 KeyboardInterrupt:

COVID-19 sequences from each Admn2 region UK




Other results modules for UK introductions can be added in here if required.

## Appendix

The plot below shows the number of sequences from each country that don't have specific enough location data to plot on the map.