Supplementary Information

Manuscript: E-government service satisfaction and citizen trust in China: Mechanisms and digital-divide boundaries

Submission ID: 070849e3-a525-4808-85c9-1b818f0d0118

This file has three parts. Part S1 gives the full variable-definition and coding scheme. Part S2 gives the numerical output behind every result reported in the manuscript, one file per analysis. Part S3 gives the complete analysis code, which states for each script what it computes and which output file it writes.

The CSGS2024 data are held by the survey sponsor under a restricted data-use agreement, which prohibits redistribution to any third party. No individual-level data are therefore included, and the code cannot be executed without first obtaining the data; the manuscript's Data availability statement gives the access route.

S1. Variable definitions and coding scheme

> Manuscript: *E-government service satisfaction and citizen trust in China: Mechanisms and digital-divide boundaries*

> Data source: China Social Governance Survey 2024 (CSGS2024; Authorization Number CSGS7054829146)

> Purpose: to enable editors and referees to verify the construction of every variable used in the analysis.

1. Analytic sample

- N = 3,694 respondents, aged 18 years and above, across 31 provincial-level units of China.
- Sampling: quota sampling anchored to the provincial distribution of household sizes (China's Seventh National Population Census, 2020); face-to-face interviews supplemented by supervised online sessions.
- Unit of analysis: individual.

2. Constructs and item mapping

All Likert items are scored 1–5. Composite scores are the arithmetic mean of the listed items.

2.1 E-government satisfaction (`EGOV_SAT_score`)

Item	CSGS variable	Content
1	`nq4_1`	Satisfaction with online government services
2	`nq4_2`	Satisfaction with online consultation / complaint reporting
3	`nq4_3`	Satisfaction with government information disclosure
4	`nq4_4`	Satisfaction with government new-media accounts
5	`nq4_5`	Satisfaction with the local government website

Scale: 1 = very dissatisfied … 5 = very satisfied. Cronbach's α = .886 (N = 3,694).

2.2 Government trust (`GOV_TRUST_score`)

Item	CSGS variable	Content
1	`gq1_1`	Trust in central government
2	`gq1_2`	Trust in provincial government
3	`gq1_3`	Trust in city/district government
4	`gq1_4`	Trust in county government
5	`gq1_5`	Trust in township government

Scale: 1 = strongly distrust … 5 = strongly trust. Cronbach's α = .919.

2.3 Perceived governance performance (`PERF_EVAL_score`)

Eight items (`hq4_1` … `hq4_8`) evaluating local government work on: crime suppression, rumour management, telecom-fraud enforcement, legal education, dispute mediation, legal-aid services, drinking-water safety, and electricity supply.

Scale: 1 = very dissatisfied … 5 = very satisfied. Cronbach's α = .902.

2.4 Digital governance belief (`DIG_ATT_score`)

- Single item `nq6_1`: agreement that government digital transformation improves administrative efficiency.
- Scale: 1 = strongly disagree … 5 = strongly agree.
- **Exclusion note:** a second available item (`nq6_3`) referred directly to trust in government; because its wording overlapped with the outcome measure, it was excluded from the mediator (avoids criterion contamination). A third item concerned privacy risk and was not used (negative correlation with the efficiency item).

2.5 Digital awareness (moderator) (`nq8`)

- Single item: self-rated familiarity with digital community services.
- Original 5-point categorical labels mapped to 1–5: 完全不了解=1, 比较不了解=2, 一般=3, 比较了解=4, 非常了解=5.

2.6 Age (moderator) (`age`)

- Derived from reported birth year: age = 2024 − birth year. Range in sample: 18–93 (M = 34.5, SD = 14.2).

2.7 Controls

Variable	CSGS variable	Coding
Gender (`male`)	`cq1`	男 = 1, 女 = 0
Age (`age`)	`cq2`	years (2024 − birth year)
Education (`edu`)	`cq3`	0 = no formal education; 1 = primary; 2 = junior secondary; 3 = senior secondary/vocational; 4 = junior college; 5 = undergraduate; 6 = postgraduate
Urban hukou (`hukou_urban`)	`fq4`	城镇户口 = 1, 农村户口 = 0

2.8 Additional variables used in robustness checks

- `nq10`: willingness to participate in digital-skills training (不愿意 = 0, 愿意 = 1).
- `sq17_1` … `sq17_6`: supplementary trust items by government level. They are carried in the analytic dataset but are not used in any reported analysis; the central-versus-local trust disaggregation in Section 4.6 uses the main trust items `gq1_1` (central government) and `gq1_5` (township government).

---

3. Data transformations

1. Likert items retained on their original 1–5 metric; no reverse-coding was required.
2. Composite scores = mean of non-missing items (listwise per construct).
3. `age` computed as 2024 − reported birth year; respondents with missing birth year excluded from age-based analyses.
4. Moderators (`nq8`, `age`) centred/standardised where the model called for it (see analysis code).
5. Post-stratification weight: inverse of the sample's own gender × age-group × urban-location cell frequency, normalised to mean 1 (documented as a diagnostic; see manuscript §3.1/§4.6).

---

4. Measurement and modelling summary

- Confirmatory factor analysis of the three multi-item scales (EGOV_SAT, PERF_EVAL, GOV_TRUST): three-factor model CFI = .887, TLI = .869, RMSEA = .104, SRMR = .044; bifactor model, with the general factor specified as orthogonal to each specific factor (the identification condition for a bifactor model), CFI = .937, TLI = .917, RMSEA = .082, SRMR = .097, AIC = 126,768, BIC = 127,104.
- Main analysis: **bifactor structural equation model** with a general evaluative factor G (orthogonal to the specific factors) plus three specific factors; mediation (EGOV → PERF / DIG → GOV_TRUST) and the direct path estimated simultaneously with controls. Standardised estimates: total β = .437, indirect via PERF β = .238, indirect via DIG β = .027, direct β = .173. Three-factor SEM without the general factor, for comparison: CFI = .882, TLI = .863, RMSEA = .085, SRMR = .040.
- Bootstrap: 1,000 resamples, bias-corrected confidence intervals; `set.seed(42)` is set in every script that bootstraps, so the intervals are reproducible.
- Software: R 4.6.1 with lavaan 0.7-2 for the latent-variable models; Python 3.14 with statsmodels 0.14.6, pandas 3.0.3, numpy 2.4.6 and scikit-learn 1.8.0 for the descriptive, OLS, weighting and figure code.

---

5. Reproducibility

All analysis scripts are provided in `analysis_code/`, with a table mapping each script to the output files it writes, and the numerical output behind every result reported in the manuscript is provided in `analysis_output/`. Every script that bootstraps sets `set.seed(42)` and uses 1,000 resamples, and no output file contains stored constants, so the reported estimates and bias-corrected confidence intervals reproduce exactly from the analytic dataset.

The CSGS2024 data are held by the survey sponsor under a restricted data-use agreement, which prohibits redistribution to any third party. Neither the raw survey file nor the derived analytic dataset can therefore be included here, and the scripts cannot be executed without first obtaining the data. Qualified researchers, including the journal's referees, may apply to the data provider through the CSGS project's designated access procedure (see the manuscript's Data availability statement).

S2. Numerical outputs

Each subsection corresponds to one output file in `analysis_output/`. Estimates are given to three decimal places; p values below the resolution of the bootstrap are reported as `0`. Output from earlier analysis rounds that does not back any reported result is held separately in the submission package and is not reproduced here.

S2.1 Sample profile (`sample_profile.csv`)

n	age_mean	age_sd	age_min	age_max	pct_aged_60_plus	pct_male	pct_urban_hukou	edu_mean
3694	34.49	14.19	18	93	4.76	46.67	55.01	4.085

Note. Backs the sample description in Sections 3.1 and 5.7. `pct_aged_60_plus` is the share of respondents aged 60 or over.

S2.2 Descriptive statistics (`descriptives.csv`)

variable	mean	std	min	median	max	n
EGOV_SAT_score	3.576	.682	1.0	3.6	5.0	3694
PERF_EVAL_score	3.748	.697	1.0	3.875	5.0	3694
GOV_TRUST_score	4.17	.769	1.0	4.2	5.0	3694
DIG_ATT_score	3.887	.751	1.0	4.0	5.0	3694
age	34.494	14.186	18.0	29.0	93.0	3694
edu	4.085	1.415	0	5.0	6.0	3694
male	.467	.499	0	0	1.0	3694
hukou_urban	.550	.498	0	1.0	1.0	3694

Note. Distribution of every analysis variable in the analytic sample (N = 3,694).

S2.3 Scale reliability (`scale_reliability.csv`)

construct	n_items	n	cronbach_alpha	source
EGOV_SAT	5	3694	.886	Module N4: government websites and informatisation
GOV_TRUST	5	3694	.919	Module G1: institutional trust, government
PERF_EVAL	8	3694	.902	Module H4: evaluation of government work

Note. Cronbach's alpha for each multi-item scale. `source` identifies the CSGS2024 module from which the items are drawn.

S2.4 Bivariate correlations (`correlation_matrix.csv`)

Variable	EGOV_SAT	PERF_EVAL	GOV_TRUST	DIG_ATT	Age	Edu	Urban
EGOV_SAT	1.0	.531	.429	.435	-.041	.046	.073
PERF_EVAL	.531	1.0	.526	.297	-.001	.019	.045
GOV_TRUST	.429	.526	1.0	.255	.034	.012	-.002
DIG_ATT	.435	.297	.255	1.0	-.026	.089	.082
Age	-.041	-.001	.034	-.026	1.0	-.567	.038
Edu	.046	.019	.012	.089	-.567	1.0	.292
Urban	.073	.045	-.002	.082	.038	.292	1.0

Note. Pearson correlations between the study variables, reported in Table 2.

S2.5 Three-factor confirmatory factor analysis (`cfa_three_factor_fit.csv`)

metric	value
chisq	5375.543
df	132
cfi	.887
tli	.869
rmsea	.104
srmr	.044
ave_egov	.609
ave_trust	.714
ave_perf	.540

Note. Fit indices and average variance extracted for the measurement model reported in Section 4.1. `ave_*` are the average variances extracted for the three constructs.

S2.6 Measurement model comparison: three-factor versus bifactor (`cfa_model_compare.csv`)

model	CFI	TLI	RMSEA	SRMR	df
Three-factor	.887	.869	.104	.044	132
Bifactor	.937	.917	.082	.097	117

Note. The two measurement models compared in Section 4.1. `df`: degrees of freedom; `CFI`: comparative fit index; `TLI`: Tucker–Lewis index; `RMSEA`: root mean square error of approximation; `SRMR`: standardised root mean square residual.

S2.7 Standardised loadings on the general factor (`cfa_bifactor_loadings.csv`)

construct	rhs	est.std
EGOV_SAT	nq4_1	.337
EGOV_SAT	nq4_2	.366
EGOV_SAT	nq4_3	.391
EGOV_SAT	nq4_4	.337
EGOV_SAT	nq4_5	.402
PERF_EVAL	hq4_1	.448
PERF_EVAL	hq4_2	.423
PERF_EVAL	hq4_3	.335
PERF_EVAL	hq4_4	.453
PERF_EVAL	hq4_5	.458
PERF_EVAL	hq4_6	.467
PERF_EVAL	hq4_7	.383
PERF_EVAL	hq4_8	.334
GOV_TRUST	gq1_1	.665
GOV_TRUST	gq1_2	.820
GOV_TRUST	gq1_3	.908
GOV_TRUST	gq1_4	.954
GOV_TRUST	gq1_5	.835

Note. Loadings of every item on the general evaluative factor of the bifactor measurement model. The ranges quoted in Section 4.1 are the minimum and maximum within each construct: `EGOV_SAT` .75 to .84, `GOV_TRUST` .68 to .96, and `PERF_EVAL` close to zero.

S2.8 Common-method-variance diagnostics (`summary_stats.csv`)

n	Harman_PCA_first_component	EGOV_alpha	GOV_TRUST_alpha	PERF_alpha	EGOV_AVE	TRUST_AVE	PERF_AVE	CFI	RMSEA	SRMR
3694	45.8	.886	.919	.902	.609	.714	.540	.887	.104	.044

Note. `Harman_PCA_first_component`: percentage of variance explained by the first principal component of all scale items (Harman's single-factor test); `*_AVE`: average variance extracted; `CFI`, `RMSEA`, `SRMR`: fit of the three-factor measurement model.

S2.9 Ordinary least squares regressions of government trust (`model_results.csv`)

label	var	coef	se	t	p	n	adj_r2	beta
M1_total	const	2.185	.091	24.031	1.14e-118	3694	.189	
M1_total	EGOV_SAT_score	.489	.017	29.154	2.45e-168	3694	.189	.434
M1_total	male	-.008	.023	-.361	.718	3694	.189	-.005
M1_total	age	.005	.001	4.641	3.60e-06	3694	.189	.087
M1_total	edu	.031	.011	2.890	.004	3694	.189	.056
M1_total	hukou_urban	-.084	.025	-3.359	.001	3694	.189	-.054
M2_mechanism	const	1.308	.093	14.053	9.75e-44	3694	.314	
M2_mechanism	EGOV_SAT_score	.218	.019	11.260	6.10e-29	3694	.314	.193
M2_mechanism	PERF_EVAL_score	.455	.018	25.443	1.07e-131	3694	.314	.412
M2_mechanism	DIG_ATT_score	.052	.016	3.314	.001	3694	.314	.051
M2_mechanism	male	-.044	.021	-2.103	.036	3694	.314	-.029
M2_mechanism	age	.004	.001	4.379	1.22e-05	3694	.314	.075
M2_mechanism	edu	.026	.010	2.691	.007	3694	.314	.048
M2_mechanism	hukou_urban	-.088	.023	-3.867	.000	3694	.314	-.057
M3a_EgovToPerf	const	1.710	.077	22.124	6.04e-102	3694	.285	
M3a_EgovToPerf	EGOV_SAT_score	.542	.014	38.039	2.10e-267	3694	.285	.531
M3a_EgovToPerf	male	.080	.019	4.132	3.68e-05	3694	.285	.058
M3a_EgovToPerf	age	.001	.001	1.311	.190	3694	.285	.023
M3a_EgovToPerf	edu	.004	.009	.494	.621	3694	.285	.009
M3a_EgovToPerf	hukou_urban	.007	.021	.317	.751	3694	.285	.005
M3b_EgovToDigAtt	const	1.921	.088	21.720	1.56e-98	3694	.195	
M3b_EgovToDigAtt	EGOV_SAT_score	.475	.016	29.073	1.69e-167	3694	.195	.431
M3b_EgovToDigAtt	male	-.009	.022	-.402	.688	3694	.195	-.006
M3b_EgovToDigAtt	age	.002	.001	2.082	.037	3694	.195	.039
M3b_EgovToDigAtt	edu	.045	.010	4.341	1.46e-05	3694	.195	.084
M3b_EgovToDigAtt	hukou_urban	.036	.024	1.487	.137	3694	.195	.024
M_urban	const	1.238	.132	9.383	1.65e-20	2032	.327	
M_urban	EGOV_SAT_score	.246	.026	9.602	2.22e-21	2032	.327	.220
M_urban	PERF_EVAL_score	.453	.024	18.681	5.61e-72	2032	.327	.406
M_urban	DIG_ATT_score	.038	.021	1.791	.073	2032	.327	.036
M_urban	male	-.054	.028	-1.896	.058	2032	.327	-.035
M_urban	age	.005	.001	4.299	1.79e-05	2032	.327	.091
M_urban	edu	.008	.014	.536	.592	2032	.327	.011
M_rural	const	1.380	.144	9.588	3.15e-21	1662	.301	
M_rural	EGOV_SAT_score	.181	.029	6.167	8.71e-10	1662	.301	.159
M_rural	PERF_EVAL_score	.460	.026	17.374	3.06e-62	1662	.301	.421
M_rural	DIG_ATT_score	.066	.023	2.899	.004	1662	.301	.067
M_rural	male	-.033	.032	-1.037	.300	1662	.301	-.021
M_rural	age	.003	.002	1.559	.119	1662	.301	.046
M_rural	edu	.033	.015	2.265	.024	1662	.301	.067

Note. The models reported in Table 3, with the urban and rural subsample estimates discussed in Section 4.6. `label`: model; `var`: predictor; `coef`: unstandardised coefficient; `se`: robust standard error; `t`: t statistic; `p`: two-tailed p value; `n`: observations; `adj_r2`: adjusted R squared. `hukou_urban` is constant within the rural subsample, so the rural model omits it.

S2.10 Fit of the bifactor structural equation model (`bifactor_sem_fit.csv`)

metric	value
chisq	2803.643
df	194
cfi	.945
tli	.930
rmsea	.060
srmr	.033

Note. Fit indices for the model that produces the study's main estimates (Section 4.4). `chisq`: model chi-square.

S2.11 Main results: bifactor structural equation model with mediation (`bifactor_mediation_results.csv`)

label	est	ci.lower	ci.upper	pvalue	est.std	std.ci.lower	std.ci.upper
a1	.716	.657	.781	0	.581	.548	.614
a2	.354	.327	.385	0	.472	.439	.508
b1	.400	.345	.458	0	.409	.360	.460
b2	.091	.025	.157	.008	.056	.015	.097
c	.208	.146	.270	8.31e-11	.173	.123	.222
ind1	.286	.236	.338	0	.238	.203	.273
ind2	.032	.009	.056	.008	.027	.007	.046
total	.527	.449	.589	0	.437	.388	.478

Note. Path estimates with 1,000 bias-corrected bootstrap resamples (`set.seed(42)`). `a1`: EGOV to PERF; `a2`: EGOV to DIG; `b1`: PERF to TRUST; `b2`: DIG to TRUST; `c`: direct EGOV to TRUST; `ind1`: indirect effect via PERF; `ind2`: indirect effect via DIG; `total`: total association. EGOV = e-government satisfaction; PERF = perceived governance performance; DIG = digital governance belief; TRUST = government trust. `est` is the standardised estimate; `ci.lower` and `ci.upper` are the 95 per cent confidence limits.

S2.12 Decomposition of the total association (`mediation_decomposition.csv`)

effect	est	pct
Total	.437	100.0
Direct (c)	.173	39.6
Indirect via PERF	.238	54.5
Indirect via DIG	.027	6.2

Note. `pct` is each component's share of the total association, computed from the reported coefficients, so the shares can be checked by hand. The digital-governance-belief pathway is small but statistically reliable (see `bifactor_mediation_results.csv`), and the shares may not sum to exactly 100 because of rounding.

S2.13 Three-factor structural equation model, for comparison (`three_factor_sem_fit.csv`)

metric	value
chisq	5820.465
df	212
cfi	.882
tli	.863
rmsea	.085
srmr	.040

Note. Fit of the same mediation model without the general evaluative factor; the comparison quoted in Section 4.4.

S2.14 Moderation of the satisfaction–trust association (`moderation_results.csv`)

moderator	interaction_B	SE	p	signif	direction	n
Digital awareness (nq8)	-.036	.010	.000	p<.001	- (weakens)	3694
Digital training will (nq10)	.014	.028	.618	n.s.	+ (strengthens)	3694
Age	-.024	.012	.039	p<.05	- (weakens)	3694
Education	.019	.012	.106	n.s.	+ (strengthens)	3694

Note. Interaction terms estimated on observed composite scores: digital awareness, willingness to take digital-skills training, age, and education. `interaction_B`: coefficient on the satisfaction-by-moderator product; `direction` summarises the sign.

S2.15 Moderated mediation: does digital awareness condition the pathways? (`moderated_mediation_results.csv`)

label	est	ci.lower	ci.upper	pvalue
a0	.510	.481	.540	0
a2	.002	-.024	.030	.904
c0	.193	.155	.226	0
b1	.410	.377	.447	0
b2	-.004	-.039	.031	.826
c1	-.045	-.081	-.013	.010
ind_mean	.209	.189	.232	0
ind_low	.210	.182	.244	0
ind_high	.207	.180	.239	0
mm_diff	-.003	-.048	.041	.897

Note. Digital awareness is entered as a moderator of the performance-mediated and direct pathways separately. `a2`: moderation of EGOV to PERF; `b2`: moderation of PERF to TRUST; `c1`: moderation of the direct EGOV to TRUST path; `ind_low`, `ind_high`: conditional indirect effects at low and high digital awareness; `mm_diff`: the moderated-mediation difference.

S2.16 Split-sample estimates plotted in Figure 3 (`figure3_slopes.csv`)

digital_awareness	n	b	se	ci_lower	ci_upper	p
1	450	.507	.056	.398	.616	0
2	780	.527	.037	.454	.600	0
3	1766	.435	.026	.384	.486	0
4	614	.365	.040	.287	.443	0
5	84	.296	.097	.106	.485	.003

Note. Satisfaction–trust association estimated separately within each level of digital awareness (1 = lowest, 5 = highest), with 95 per cent confidence limits. These are the values plotted in Figure 3.

S2.17 Trust and satisfaction by level of digital awareness (`group_trust_distribution.csv`)

digaware_level	n	trust_m	trust_sd	egov_m	egov_sd
1	450	3.846	.923	3.221	.717
2	780	4.086	.784	3.471	.675
3	1766	4.183	.728	3.562	.619
4	614	4.416	.629	3.894	.604
5	84	4.624	.678	4.421	.750

Note. `digaware_level` runs from 1 (lowest digital awareness) to 5 (highest). `trust_m`, `trust_sd`: mean and standard deviation of government trust; `egov_m`, `egov_sd`: the same for e-government satisfaction.

S2.18 Government trust disaggregated by level of government (`trust_level_robustness.csv`)

outcome	outcome_label	model	predictor	B	SE	p	n
central	Trust in central government (gq1_1)	total	EGOV_SAT	.346	.017	0	3694
central	Trust in central government (gq1_1)	mechanism	EGOV_SAT	.159	.020	0	3694
central	Trust in central government (gq1_1)	mechanism	PERF_EVAL	.345	.019	0	3694
local	Trust in township government (gq1_5)	total	EGOV_SAT	.594	.023	0	3694
local	Trust in township government (gq1_5)	mechanism	EGOV_SAT	.286	.025	0	3694
local	Trust in township government (gq1_5)	mechanism	PERF_EVAL	.566	.024	0	3694

Note. The same models estimated separately on trust in central government (`gq1_1`) and trust in township government (`gq1_5`), as reported in Section 4.6. `model` distinguishes the total-effect from the mechanism specification.

S2.19 Sensitivity of the belief pathway to measurement error (`dig_reliability_sensitivity.csv`)

reliability	a2	b2	indirect_DIG	ind2_p	total
1	.472	.056	.027	.001	.437
.800	.528	.068	.036	.001	.437
.700	.565	.077	.044	.001	.437
.600	.610	.091	.056	.001	.437

Note. The single-item belief mediator is re-specified as a latent variable whose residual variance corresponds to reliabilities of .80, .70 and .60, alongside the error-free case (reliability 1.0). `indirect_DIG` is the standardised indirect effect through digital governance belief and `ind2_p` its p value; the estimate rises as assumed reliability falls, indicating that the reported estimate is attenuated rather than inflated by measurement error.

S2.20 Post-stratification weighting diagnostic (`weighting_comparison.csv`)

spec	EGOV_b	PERF_b	DIG_b	adj_R2
unweighted	.218	.455	.052	.314
weighted	.190	.475	.060	.324

Note. Main coefficients estimated without and with the approximate post-stratification weight. Weighting attenuates the satisfaction coefficient modestly and leaves the pattern of results unchanged.

S3. Analysis code

The scripts are listed in execution order and reproduce every estimate reported in the manuscript from the analytic dataset. Because that dataset cannot be redistributed, they cannot be run without first obtaining the data through the access route given in the manuscript's Data availability statement.

S3.1 `01_preprocess.py`

```
#!/usr/bin/env python3
"""Recode the CSGS2024 survey file into the analytic datasets.

This script recodes the raw survey file into the analytic dataset used by every
downstream script, derives the construct scores, and computes scale reliability.

Supports: Table 1 (descriptive statistics, via 02_models.py) and the reliability
column of Supplementary S1.

Reads:  $DATA/<raw survey file>
Writes: $DATA/analysis_dataset.csv, $DATA/item_level.csv,
        $OUT/scale_reliability.csv, $OUT/sample_profile.csv

Item-to-construct mapping:
  E-government satisfaction EGOV_SAT (nq4_1..nq4_5)
  Government trust GOV_TRUST (gq1_1..gq1_5, central to township government)
  Perceived governance performance (hq4_1..hq4_8)
  Digital governance belief (nq6_1 only; nq6_3 excluded, see below)
  Online/platform participation (sq17_1..sq17_6)
"""

import os
from pathlib import Path

# Paths are configurable so that the scripts run anywhere. Defaults assume the
# package layout: <package>/analysis_code, <package>/analysis_output,
# <package>/figures, and the restricted analytic datasets in <package>/data.
DATA = Path(os.environ.get("CSGS_DATA_DIR", "data"))
OUT = Path(os.environ.get("CSGS_OUT_DIR", "analysis_output"))
SUP = Path(os.environ.get("CSGS_SUPERSEDED_DIR", "superseded_specifications"))
FIG = Path(os.environ.get("CSGS_FIG_DIR", "figures"))
import os, numpy as np, pandas as pd
from pathlib import Path

RAW = Path(os.environ.get("CSGS_RAW_FILE",
                          "CSGS2024PubV20250309.dta"))
DATA.mkdir(parents=True, exist_ok=True)
OUT.mkdir(parents=True, exist_ok=True)

df = pd.read_stata(RAW, convert_categoricals=False, convert_dates=False)

# Likert response mappings
LIK = {'非常不满意':1,'比较不满意':2,'说不上满意不满意':3,'比较满意':4,'非常满意':5}
LIK_TRUST = {'非常不信任':1,'比较不信任':2,'说不上信任不信任':3,'比较信任':4,'非常信任':5}
AGREE = {'非常不同意':1,'比较不同意':2,'说不上同意不同意':3,'比较同意':4,'非常同意':5}

def encode(series, mapping):
    return series.map(mapping)

# E-government satisfaction: nq4_1..nq4_5
egov_items = ['nq4_1','nq4_2','nq4_3','nq4_4','nq4_5']
for v in egov_items:
    df[v+'_s'] = df[v].map(LIK)

# Government trust: gq1_1..gq1_5 (five levels of government)
trust_items = ['gq1_1','gq1_2','gq1_3','gq1_4','gq1_5']
for v in trust_items:
    df[v+'_s'] = df[v].map(LIK_TRUST)

# Perceived governance performance: hq4_1..hq4_8
perf_items = ['hq4_1','hq4_2','hq4_3','hq4_4','hq4_5','hq4_6','hq4_7','hq4_8']
for v in perf_items:
    df[v+'_s'] = df[v].map(LIK)

# --- Digital governance belief ---
# Item nq6_1 (belief that digital transformation improves administrative
# efficiency) is used on its own. Item nq6_3 is excluded because its wording
# refers to trust in government and would overlap with the outcome measure.
df['nq6_1_s'] = df['nq6_1'].map(AGREE)

# ---------- Cronbach alpha ----------
def cronbach(items_df):
    mat = items_df.values.astype(float)
    k = mat.shape[1]
    if any(np.isnan(mat.sum(axis=1))) or mat.shape[0] < k:
        return np.nan
    var_total = mat.sum(axis=1).var(ddof=1)
    var_items = [mat[:,i].var(ddof=1) for i in range(k)]
    return k/(k-1)*(1 - sum(var_items)/var_total)

alpha_res = []
for name, items, loc in [
    ("EGOV_SAT", egov_items, "Module N4: government websites and informatisation"),
    ("GOV_TRUST", trust_items, "Module G1: institutional trust, government"),
    ("PERF_EVAL", perf_items, "Module H4: evaluation of government work"),
]:
```

```
    sub = df[[v+'_s' for v in items]].dropna()
    alpha_res.append({
        "construct": name, "n_items": len(items), "n": len(sub),
        "cronbach_alpha": round(cronbach(sub),3), "source": loc
    })
alpha_df = pd.DataFrame(alpha_res)
alpha_df.to_csv(OUT/"scale_reliability.csv", index=False)
print(alpha_df.to_string(index=False))

# Correlations between construct scores
# Composite scores: mean of the non-missing items
for name, items in [("EGOV_SAT",egov_items),("GOV_TRUST",trust_items),("PERF_EVAL",perf_items)]:
    df[f"{name}_score"] = df[[v+'_s' for v in items]].mean(axis=1)
df['DIG_ATT_score'] = df['nq6_1_s']  # single item; nq6_3 excluded (overlaps the outcome)

corr_cols = ['EGOV_SAT_score','GOV_TRUST_score','PERF_EVAL_score','DIG_ATT_score']
print("\n=== Construct correlations ===")
print(df[corr_cols].corr().round(3).to_string())

# Demographic controls
# Gender
df['male'] = df['cq1'].map({'男':1,'女':0})
# Age, derived from the reported birth year
def to_year(v):
    if pd.isna(v): return np.nan
    try:
        return pd.to_datetime(v).year
    except Exception:
        try: return int(float(v))
        except Exception: return np.nan
df['birthyear'] = df['cq2'].apply(to_year)
df['age'] = df['birthyear'].apply(lambda y: (2024 - y) if pd.notna(y) else np.nan)
# Education (0 = no formal education, 6 = postgraduate)
edu_map = {'未接受过正式教育':0,'小学':1,'初中':2,'高中或中专':3,'大专或高职':4,'本科':5,'研究生及以上':6}
df['edu'] = df['cq3'].map(edu_map)
# Hukou type (urban/rural)
df['hukou_type'] = df['fq4'] if 'fq4' in df.columns else np.nan
# Ethnicity (Han = 1); the field may be a string or a code
def eth(x):
    s = str(x)
    if s.startswith('汉'): return 1
    return 0 if s not in ('nan','') else np.nan
df['ethnic'] = df['cq9'].apply(eth) if 'cq9' in df.columns else np.nan

# Save the analysis columns
keep = ['EGOV_SAT_score','GOV_TRUST_score','PERF_EVAL_score','DIG_ATT_score',
        'male','age','edu','hukou_type','nq8','nq9','nq10',
        'sq17_1','sq17_2','sq17_3','sq17_4','sq17_5','sq17_6']
keep = [k for k in keep if k in df.columns]
out_df = df[keep].copy()
# Put the composite scores first
front = [k for k in ['EGOV_SAT_score','GOV_TRUST_score','PERF_EVAL_score','DIG_ATT_score'] if k in out_df.columns]
out_df = out_df[front + [k for k in keep if k not in front]]
out_df.to_csv(DATA/"analysis_dataset.csv", index=False)
print("\nAnalytic dataset saved to", DATA / "analysis_dataset.csv",
      "shape:", out_df.shape)
print("age non-missing:", out_df["age"].notna().sum(),
      " hukou_type non-missing:", out_df["hukou_type"].notna().sum())

# --- Item-level file used by the measurement and latent-variable models ---
df['nq6_3_s'] = df['nq6_3'].map(AGREE)  # carried for completeness; not used in the analysis
item_cols = egov_items + trust_items + perf_items + ['nq6_1', 'nq6_3']
items = df[[c + '_s' for c in item_cols]].copy()
items.columns = item_cols
items['male'] = df['male']
items['edu'] = df['edu']
items['age'] = df['age']
items.to_csv(DATA / "item_level.csv", index=False)
print("Item-level dataset saved to", DATA / "item_level.csv", "shape:", items.shape)

# --- Sample profile: backs the sample description in Sections 3.1 and 5.7 ---
profile = pd.DataFrame([{
    "n": int(len(out_df)),
    "age_mean": round(float(out_df["age"].mean()), 2),
    "age_sd": round(float(out_df["age"].std()), 2),
    "age_min": int(out_df["age"].min()),
    "age_max": int(out_df["age"].max()),
    "pct_aged_60_plus": round(100 * float((out_df["age"] >= 60).mean()), 2),
    "pct_male": round(100 * float(out_df["male"].mean()), 2),
    "pct_urban_hukou": round(100 * float((out_df["hukou_type"] == "\u57ce\u9547\u6237\u53e3").mean()), 2),
    "edu_mean": round(float(out_df["edu"].mean()), 3),
}])
profile.to_csv(OUT / "sample_profile.csv", index=False)
print("Sample profile saved ->", OUT / "sample_profile.csv")
```

S3.2 `02_models.py`

```
#!/usr/bin/env python3
"""Ordinary least squares models of government trust.

Supports Table 1 (descriptive statistics) and Table 3 (OLS regressions) of the
manuscript, and the urban/rural subsample estimates reported in Section 4.6.

Models, all on observed composite scores:
  M1  GOV_TRUST ~ EGOV_SAT + controls                 total association
  M2  + PERF_EVAL + DIG_ATT                           mechanism decomposition
  M3a/M3b  the two mechanism equations on their own
  Urban / rural subsample replications of M2

Reads:  $DATA/analysis_dataset.csv
Writes: $OUT/descriptives.csv, $OUT/model_results.csv,
        $SUP/observed_score_decomposition.csv
"""

import os
from pathlib import Path

# Paths are configurable so that the scripts run anywhere. Defaults assume the
# package layout: <package>/analysis_code, <package>/analysis_output,
# <package>/figures, and the restricted analytic datasets in <package>/data.
DATA = Path(os.environ.get("CSGS_DATA_DIR", "data"))
OUT = Path(os.environ.get("CSGS_OUT_DIR", "analysis_output"))
SUP = Path(os.environ.get("CSGS_SUPERSEDED_DIR", "superseded_specifications"))
FIG = Path(os.environ.get("CSGS_FIG_DIR", "figures"))
import numpy as np, pandas as pd
import statsmodels.api as sm
from pathlib import Path

DATA_FILE = DATA / "analysis_dataset.csv"

df = pd.read_csv(DATA_FILE)
# Controls
df['male'] = df['male'].astype(float)
df['age'] = df['age'].astype(float)
df['edu'] = df['edu'].astype(float)
df['hukou_urban'] = df['hukou_type'].map({'城镇户口':1,'农村户口':0})
print("=== Descriptive statistics ===")
desc_cols = ['EGOV_SAT_score','PERF_EVAL_score','GOV_TRUST_score','DIG_ATT_score',
             'age','edu','male','hukou_urban']
desc = df[desc_cols].describe().T[['mean','std','min','50%','max']].round(3)
desc['n'] = df[desc_cols].count()
print(desc.to_string())
desc.index.name = 'variable'
desc = desc.rename(columns={'50%': 'median'})
desc.to_csv(OUT / "descriptives.csv")


def run_ols(y, X_cols, data, label):
    dt = data[X_cols + [y]].dropna()
    X = sm.add_constant(dt[X_cols])
    model = sm.OLS(dt[y], X).fit()
    out = pd.DataFrame({
        'label': label,
        'var': model.params.index,
        'coef': model.params.values,
        'se': model.bse.values,
        't': model.tvalues.values,
        'p': model.pvalues.values,
    })
    out['n'] = len(dt)
    out['adj_r2'] = model.rsquared_adj
    # 标准化系数：便于与 §4.4 潜变量标准化估计直接比较
    sd_y = dt[y].std()
    out['beta'] = [round(float(model.params[v]) * dt[v].std() / sd_y, 3)
                   if v in dt.columns and dt[v].std() > 0 else np.nan
                   for v in model.params.index]
    return out

controls = ['male','age','edu','hukou_urban']

# M1: total association
m1 = run_ols('GOV_TRUST_score', ['EGOV_SAT_score'] + controls, df, "M1_total")
# M2: adds the two mechanism variables
m2 = run_ols('GOV_TRUST_score', ['EGOV_SAT_score','PERF_EVAL_score','DIG_ATT_score'] + controls, df, "M2_mechanism")
# M3a / M3b: the two mechanism equations
m3a = run_ols('PERF_EVAL_score', ['EGOV_SAT_score'] + controls, df, "M3a_EgovToPerf")
m3b = run_ols('DIG_ATT_score', ['EGOV_SAT_score'] + controls, df, "M3b_EgovToDigAtt")
# Trust disaggregated by level of government is produced by 07_robustness.py
# Urban and rural subsamples
sub_controls = ['male','age','edu']  # hukou_urban is constant within each subsample
df_urban = df[df['hukou_urban']==1]
df_rural = df[df['hukou_urban']==0]
m_u = run_ols('GOV_TRUST_score', ['EGOV_SAT_score','PERF_EVAL_score','DIG_ATT_score']+sub_controls, df_urban, "M_urban")
```

```
m_r = run_ols('GOV_TRUST_score', ['EGOV_SAT_score','PERF_EVAL_score','DIG_ATT_score']+sub_controls, df_rural, "M_rural")

all_out = pd.concat([m1,m2,m3a,m3b,m_u,m_r], ignore_index=True)
all_out.to_csv(OUT / "model_results.csv", index=False)

print("\n===== M2 mechanism model =====")
print(all_out[all_out.label=='M2_mechanism'][['var','coef','se','p']].to_string(index=False))
print("\n===== M1 total association =====")
print(all_out[all_out.label=='M1_total'][['var','coef','se','p']].to_string(index=False))
print("\nResults saved to", OUT / "model_results.csv")


# --- Observed-composite decomposition of the OLS coefficients ---
# Algebraic decomposition implied by the observed-score models above. It is not the
# specification reported in the manuscript, which models a general evaluative factor
# (see 18_bifactor_sem_mediation.R); it is kept in superseded_specifications/ so the
# intermediate specification remains auditable.
def coef(frame, var):
    return float(frame.loc[frame['var'] == var, 'coef'].iloc[0])

_c = coef(m2, 'EGOV_SAT_score')
_b1 = coef(m2, 'PERF_EVAL_score')
_b2 = coef(m2, 'DIG_ATT_score')
_a1 = coef(m3a, 'EGOV_SAT_score')
_a2 = coef(m3b, 'EGOV_SAT_score')
_ind1, _ind2 = _a1 * _b1, _a2 * _b2

observed = pd.DataFrame({
    'label': ['c', 'b1', 'b2', 'a1', 'a2', 'ind1', 'ind2', 'total'],
    'est': [_c, _b1, _b2, _a1, _a2, _ind1, _ind2, _c + _ind1 + _ind2],
})
SUP.mkdir(parents=True, exist_ok=True)
observed.to_csv(SUP / "observed_score_decomposition.csv", index=False)
print("\nObserved-score decomposition saved to",
      SUP / "observed_score_decomposition.csv")
```

S3.3 `03_cfa.R`

```
#!/usr/bin/env Rscript
# Three-factor confirmatory factor analysis of the three multi-item scales.
#
# Supports the measurement-model results reported in Section 4.1: fit indices and
# average variance extracted for AGOV_SAT, PERF_EVAL and GOV_TRUST.
#
# Reads:  $DATA/item_level.csv
# Writes: $OUT/cfa_three_factor_fit.csv

# Paths are configurable so that the scripts run anywhere. Defaults assume the
# package layout: <package>/analysis_code, <package>/analysis_output,
# <package>/figures, and the restricted analytic datasets in <package>/data.
DATA <- Sys.getenv("CSGS_DATA_DIR", "data")
OUT <- Sys.getenv("CSGS_OUT_DIR", "analysis_output")
SUP <- Sys.getenv("CSGS_SUPERSEDED_DIR", "superseded_specifications")
FIG <- Sys.getenv("CSGS_FIG_DIR", "figures")
suppressMessages(library(lavaan))

rd <- read.csv(file.path(DATA, "item_level.csv"))
cols <- c(paste0("nq4_", 1:5), paste0("gq1_", 1:5), paste0("hq4_", 1:8))
rd <- rd[complete.cases(rd[, cols]), ]
cat("CFA sample:", nrow(rd), "\n")

model <- '
  EGOV =~ nq4_1 + nq4_2 + nq4_3 + nq4_4 + nq4_5
  TRUST =~ gq1_1 + gq1_2 + gq1_3 + gq1_4 + gq1_5
  PERF =~ hq4_1 + hq4_2 + hq4_3 + hq4_4 + hq4_5 + hq4_6 + hq4_7 + hq4_8
'
fit <- cfa(model, data = rd, estimator = "MLR", std.lv = TRUE)
cat("\n===== Three-factor CFA fit =====\n")
fm <- fitMeasures(fit, c("chisq", "df", "cfi", "tli", "rmsea", "srmr"))
print(round(fm, 3))

cat("\n===== Factor correlations =====\n")
print(round(inspect(fit, "cor.ov")[1:3, 1:3], 3))

# Average variance extracted from the standardised loadings
std <- standardizedSolution(fit)
avg <- function(col) mean(std[std$op == "=~" & std$rhs %in% col, "est.std"]^2)
ave <- c(ave_egov = avg(paste0("nq4_", 1:5)),
         ave_trust = avg(paste0("gq1_", 1:5)),
         ave_perf = avg(paste0("hq4_", 1:8)))
cat("\nAVE:\n"); print(round(ave, 3))

out <- data.frame(metric = c(names(fm), names(ave)),
                  value = c(unname(fm), unname(ave)))
write.csv(out, file.path(OUT, "cfa_three_factor_fit.csv"), row.names = FALSE)
cat("\nSaved ->", file.path(OUT, "cfa_three_factor_fit.csv"), "\n")
```

S3.4 `10_cfa_compare.R`

```
#!/usr/bin/env Rscript
# Confirmatory factor analysis: three-factor versus bifactor measurement model.
#
# Supports the measurement models reported in Section 4.1, including the
# general-factor loadings quoted there.
#
# Reads:  $DATA/item_level.csv
# Writes: $OUT/cfa_model_compare.csv, $OUT/cfa_bifactor_loadings.csv

# Paths are configurable so that the scripts run anywhere. Defaults assume the
# package layout: <package>/analysis_code, <package>/analysis_output,
# <package>/figures, and the restricted analytic datasets in <package>/data.
DATA <- Sys.getenv("CSGS_DATA_DIR", "data")
OUT <- Sys.getenv("CSGS_OUT_DIR", "analysis_output")
SUP <- Sys.getenv("CSGS_SUPERSEDED_DIR", "superseded_specifications")
FIG <- Sys.getenv("CSGS_FIG_DIR", "figures")
suppressMessages(library(lavaan))

rd <- read.csv(file.path(DATA, "item_level.csv"))
cols <- c(paste0("nq4_",1:5), paste0("hq4_",1:8), paste0("gq1_",1:5))
rd <- rd[complete.cases(rd[,cols]), cols]
cat("CFA sample:", nrow(rd), "\n")

egov <- paste0("nq4_",1:5); perf <- paste0("hq4_",1:8); trust <- paste0("gq1_",1:5)

models <- list(
  three = "EGOV =~ nq4_1+nq4_2+nq4_3+nq4_4+nq4_5
           PERF =~ hq4_1+hq4_2+hq4_3+hq4_4+hq4_5+hq4_6+hq4_7+hq4_8
           TRUST =~ gq1_1+gq1_2+gq1_3+gq1_4+gq1_5"
)

fit_three <- cfa(models$three, data=rd, estimator="MLR", std.lv=TRUE)
cat("\n===== Three-factor model fit =====\n")
fm3 <- fitMeasures(fit_three, c("chisq","df","cfi","tli","rmsea","srmr"))
print(round(fm3,3))

# Bifactor: one general evaluative factor plus the three specific factors.
# The general factor is specified as orthogonal to each specific factor, which is
# the identification condition for a bifactor model; leaving those covariances free
# makes the model unidentified and its standard errors meaningless.
bifac <- '
  G =~ nq4_1+nq4_2+nq4_3+nq4_4+nq4_5+hq4_1+hq4_2+hq4_3+hq4_4+hq4_5+hq4_6+hq4_7+hq4_8+gq1_1+gq1_2+gq1_3+gq1_4+gq1_5
  EGOV =~ nq4_1+nq4_2+nq4_3+nq4_4+nq4_5
  PERF =~ hq4_1+hq4_2+hq4_3+hq4_4+hq4_5+hq4_6+hq4_7+hq4_8
  TRUST =~ gq1_1+gq1_2+gq1_3+gq1_4+gq1_5
  G ~~ 0*EGOV + 0*PERF + 0*TRUST
  EGOV ~~ 0*PERF + 0*TRUST
  PERF ~~ 0*TRUST
'
fit_bi <- cfa(bifac, data=rd, estimator="MLR", std.lv=TRUE)
cat("\n===== Bifactor model fit =====\n")
fmbi <- fitMeasures(fit_bi, c("chisq","df","cfi","tli","rmsea","srmr"))
print(round(fmbi,3))

cat("\n===== Model comparison =====\n")
cmp <- data.frame(
  model=c("Three-factor","Bifactor"),
  CFI=round(c(fm3["cfi"],fmbi["cfi"]),3), TLI=round(c(fm3["tli"],fmbi["tli"]),3),
  RMSEA=round(c(fm3["rmsea"],fmbi["rmsea"]),3), SRMR=round(c(fm3["srmr"],fmbi["srmr"]),3),
  df=c(fm3["df"],fmbi["df"]))
print(cmp)
write.csv(cmp, file.path(OUT, "cfa_model_compare.csv"), row.names = FALSE)
cat("Saved ->", file.path(OUT, "cfa_model_compare.csv"), "\n")
# Standardised loadings on the general factor, quoted as ranges in Section 4.1
std <- standardizedSolution(fit_bi)
loadings <- std[std$op == "=~" & std$lhs == "G", c("rhs", "est.std")]
loadings$construct <- ifelse(loadings$rhs %in% paste0("nq4_", 1:5), "EGOV_SAT",
                      ifelse(loadings$rhs %in% paste0("gq1_", 1:5), "GOV_TRUST", "PERF_EVAL"))
loadings$est.std <- round(loadings$est.std, 3)
write.csv(loadings[, c("construct", "rhs", "est.std")],
          file.path(OUT, "cfa_bifactor_loadings.csv"), row.names = FALSE)
print(loadings)
cat("Saved ->", file.path(OUT, "cfa_bifactor_loadings.csv"), "\n")
```

S3.5 `04_cmb.py`

```
#!/usr/bin/env python3
"""Common-method-variance diagnostics and discriminant validity.

Supports the common-method-variance checks reported in Section 4.6 and the
average-variance-extracted figures reported in Section 4.1.

Computes Harman's single-factor test on all scale items, the Fornell-Larcker
criterion for discriminant validity, and collects the reliability and
measurement-model indices into one summary table.

Reads:  $DATA/item_level.csv, $DATA/analysis_dataset.csv,
        $OUT/cfa_three_factor_fit.csv (average variance extracted)
Writes: $OUT/summary_stats.csv
"""

import os
from pathlib import Path

# Paths are configurable so that the scripts run anywhere. Defaults assume the
# package layout: <package>/analysis_code, <package>/analysis_output,
# <package>/figures, and the restricted analytic datasets in <package>/data.
DATA = Path(os.environ.get("CSGS_DATA_DIR", "data"))
OUT = Path(os.environ.get("CSGS_OUT_DIR", "analysis_output"))
SUP = Path(os.environ.get("CSGS_SUPERSEDED_DIR", "superseded_specifications"))
FIG = Path(os.environ.get("CSGS_FIG_DIR", "figures"))
import numpy as np, pandas as pd
from pathlib import Path
from sklearn.decomposition import PCA


rd = pd.read_csv(DATA / "item_level.csv")
item_cols = (list("nq4_%d"%i for i in range(1,6)) +
             list("gq1_%d"%i for i in range(1,6)) +
             list("hq4_%d"%i for i in range(1,9)))
item_cols = [c for c in item_cols if c in rd.columns]
X = rd[item_cols].dropna()
print("Common-method-variance sample:", len(X))

# Harman's single-factor test: first principal component of all scale items
pca = PCA()
pca.fit(X)
evr = pca.explained_variance_ratio_
print("\n=== Harman single-factor test ===")
print(f"Variance explained by the first principal component: {evr[0]*100:.1f}%")
print(f"First three components, cumulative: {evr[:3].sum()*100:.1f}%")
harman = evr[0]*100

# Discriminant validity: Fornell-Larcker criterion on the observed composites
av = pd.read_csv(DATA / "analysis_dataset.csv")
constructs = {
    'EGOV_SAT': ['nq4_1','nq4_2','nq4_3','nq4_4','nq4_5'],
    'GOV_TRUST': ['gq1_1','gq1_2','gq1_3','gq1_4','gq1_5'],
    'PERF_EVAL': ['hq4_1','hq4_2','hq4_3','hq4_4','hq4_5','hq4_6','hq4_7','hq4_8'],
}
scores = {}
for k, cols in constructs.items():
    scores[k] = rd[cols].mean(axis=1)
S = pd.DataFrame(scores)
corr = S.corr().round(3)
print("\n=== Construct correlation matrix ===")
print(corr.to_string())
# Average variance extracted, read from the confirmatory factor analysis output
_cfa_file = OUT / "cfa_three_factor_fit.csv"
_rel_file = OUT / "scale_reliability.csv"
if not _cfa_file.exists() or not _rel_file.exists():
    raise SystemExit("Run 01_preprocess.py and 03_cfa.R first.")
_m = pd.read_csv(_cfa_file).set_index("metric")["value"]
_rel = pd.read_csv(_rel_file).set_index("construct")["cronbach_alpha"]
ave = {"EGOV_SAT": float(_m["ave_egov"]), "GOV_TRUST": float(_m["ave_trust"]),
       "PERF_EVAL": float(_m["ave_perf"])}
# Fornell-Larcker: the square root of AVE must exceed every correlation
print("\n=== Fornell-Larcker discriminant validity (sqrt(AVE) vs correlations) ===")
for k in constructs:
    print(f"  {k}: sqrt(AVE)={np.sqrt(ave[k]):.3f}  largest correlation={corr[k].drop(k).max():.3f}")

# Collect the summary statistics
summary = {
    'n': len(rd),
    'Harman_PCA_first_component': round(harman,1),
    'EGOV_alpha': float(_rel['EGOV_SAT']), 'GOV_TRUST_alpha': float(_rel['GOV_TRUST']),
    'PERF_alpha': float(_rel['PERF_EVAL']),
    'EGOV_AVE': float(_m['ave_egov']), 'TRUST_AVE': float(_m['ave_trust']),
    'PERF_AVE': float(_m['ave_perf']),
    'CFI': float(_m['cfi']), 'RMSEA': float(_m['rmsea']), 'SRMR': float(_m['srmr']),
}
# Only measurement-level diagnostics belong here. Mediation estimates are
```

```
# reported exclusively in bifactor_mediation_results.csv and
# mediation_decomposition.csv.
pd.DataFrame([summary]).to_csv(OUT / "summary_stats.csv", index=False)
print("\n=== Summary table saved ===")
for k,v in summary.items():
    print(f"  {k}: {v}")
```

S3.6 `07_robustness.py`

```
#!/usr/bin/env python3
"""Robustness: government trust disaggregated by level of government.

Supports the central-versus-local contrast reported in Section 4.6.

Government trust is re-estimated separately on trust in central government
(gq1_1) and trust in the most local level, township government (gq1_5), using
both the total-effect and the mechanism specification.

Reads:  $DATA/analysis_dataset.csv, $DATA/item_level.csv
Writes: $OUT/trust_level_robustness.csv
"""

import os
from pathlib import Path

# Paths are configurable so that the scripts run anywhere. Defaults assume the
# package layout: <package>/analysis_code, <package>/analysis_output,
# <package>/figures, and the restricted analytic datasets in <package>/data.
DATA = Path(os.environ.get("CSGS_DATA_DIR", "data"))
OUT = Path(os.environ.get("CSGS_OUT_DIR", "analysis_output"))
SUP = Path(os.environ.get("CSGS_SUPERSEDED_DIR", "superseded_specifications"))
FIG = Path(os.environ.get("CSGS_FIG_DIR", "figures"))
import numpy as np
import pandas as pd
import statsmodels.api as sm

ds = pd.read_csv(DATA / "analysis_dataset.csv")
rd = pd.read_csv(DATA / "item_level.csv")

TRUST = {'\u975e\u5e38\u4e0d\u4fe1\u4efb': 1, '\u6bd4\u8f83\u4e0d\u4fe1\u4efb': 2,
         '\u8bf4\u4e0d\u4e0a\u4fe1\u4efb\u4e0d\u4fe1\u4efb': 3,
         '\u6bd4\u8f83\u4fe1\u4efb': 4, '\u975e\u5e38\u4fe1\u4efb': 5}

d = pd.DataFrame({
    'EGOV_SAT': ds.EGOV_SAT_score,
    'PERF_EVAL': ds.PERF_EVAL_score,
    'male': ds.male, 'age': ds.age, 'edu': ds.edu,
    'hukou_urban': ds.hukou_type.map({'\u57ce\u9547\u6237\u53e3': 1, '\u519c\u6751\u6237\u53e3': 0}).astype(float),
})

def recode(col):
    s = rd[col]
    return s.map(TRUST) if s.dtype == object else s

outcomes = [('central', 'Trust in central government (gq1_1)', recode('gq1_1')),
            ('local', 'Trust in township government (gq1_5)', recode('gq1_5'))]

rows = []
for key, label, series in outcomes:
    d['OUTCOME'] = series
    for spec, predictors in [('total', ['EGOV_SAT']),
                             ('mechanism', ['EGOV_SAT', 'PERF_EVAL'])]:
        cols = predictors + ['male', 'age', 'edu', 'hukou_urban']
        dd = d[cols + ['OUTCOME']].dropna()
        m = sm.OLS(dd['OUTCOME'], sm.add_constant(dd[cols])).fit()
        for v in predictors:
            rows.append({'outcome': key, 'outcome_label': label, 'model': spec,
                         'predictor': v, 'B': round(float(m.params[v]), 3),
                         'SE': round(float(m.bse[v]), 3),
                         'p': round(float(m.pvalues[v]), 4),
                         'n': int(m.nobs)})
        print(f"{label} [{spec}] n={int(m.nobs)} "
              f"EGOV B={m.params['EGOV_SAT']:.3f} p={m.pvalues['EGOV_SAT']:.4f}")

res = pd.DataFrame(rows)
res.to_csv(OUT / "trust_level_robustness.csv", index=False)
print("\nSaved ->", OUT / "trust_level_robustness.csv")
```

S3.7 `08_corr.py`

```
#!/usr/bin/env python3
"""Bivariate correlation matrix for the study variables.

Supports Table 2 of the manuscript.

Reads:  $DATA/analysis_dataset.csv
Writes: $OUT/correlation_matrix.csv
"""

import os
from pathlib import Path

# Paths are configurable so that the scripts run anywhere. Defaults assume the
# package layout: <package>/analysis_code, <package>/analysis_output,
# <package>/figures, and the restricted analytic datasets in <package>/data.
DATA = Path(os.environ.get("CSGS_DATA_DIR", "data"))
OUT = Path(os.environ.get("CSGS_OUT_DIR", "analysis_output"))
SUP = Path(os.environ.get("CSGS_SUPERSEDED_DIR", "superseded_specifications"))
FIG = Path(os.environ.get("CSGS_FIG_DIR", "figures"))
import pandas as pd

df = pd.read_csv(DATA / "analysis_dataset.csv")
df['hukou_urban'] = df['hukou_type'].map({'\u57ce\u9547\u6237\u53e3': 1, '\u519c\u6751\u6237\u53e3': 0})

cols = ['EGOV_SAT_score', 'PERF_EVAL_score', 'GOV_TRUST_score', 'DIG_ATT_score',
        'age', 'edu', 'hukou_urban']
labels = ['EGOV_SAT', 'PERF_EVAL', 'GOV_TRUST', 'DIG_ATT', 'Age', 'Edu', 'Urban']

c = df[cols].corr().round(3)
c.index = labels
c.columns = labels
c.to_csv(OUT / "correlation_matrix.csv")
print(c.to_string())
print("\nSaved ->", OUT / "correlation_matrix.csv")
```

S3.8 `09_moderation.py`

```
#!/usr/bin/env python3
"""Moderation of the satisfaction–trust association.

Supports the moderation results in Section 4.5 and the digital-awareness
interaction reported in the Abstract.

Model: GOV_TRUST ~ EGOV_SAT * MODERATOR + PERF_EVAL + controls.
Moderators tested: digital awareness (nq8), willingness to take digital-skills
training (nq10), age, and education.

Reads:  $DATA/analysis_dataset.csv
Writes: $OUT/moderation_results.csv
"""

import os
from pathlib import Path

# Paths are configurable so that the scripts run anywhere. Defaults assume the
# package layout: <package>/analysis_code, <package>/analysis_output,
# <package>/figures, and the restricted analytic datasets in <package>/data.
DATA = Path(os.environ.get("CSGS_DATA_DIR", "data"))
OUT = Path(os.environ.get("CSGS_OUT_DIR", "analysis_output"))
SUP = Path(os.environ.get("CSGS_SUPERSEDED_DIR", "superseded_specifications"))
FIG = Path(os.environ.get("CSGS_FIG_DIR", "figures"))
import numpy as np, pandas as pd
import statsmodels.api as sm
from pathlib import Path

df = pd.read_csv(DATA / "analysis_dataset.csv")
df['hukou_urban'] = df['hukou_type'].map({'城镇户口':1,'农村户口':0})
controls=['male','age','edu','hukou_urban']

# Code the moderators
df['digaware'] = df['nq8'].map({'完全不了解':1,'比较不了解':2,'一般':3,'比较了解':4,'非常了解':5})
df['digtrain'] = df['nq10'].map({'不愿意':0,'愿意':1})
df['age_c'] = (df['age'] - df['age'].mean())/df['age'].std()
df['edu_c'] = (df['edu'] - df['edu'].mean())/df['edu'].std()

def mod_model(mod, df, label):
    d = df[['GOV_TRUST_score','EGOV_SAT_score','PERF_EVAL_score',mod,'male','age','edu','hukou_urban']].dropna()
    d['egov'] = (d['EGOV_SAT_score'] - d['EGOV_SAT_score'].mean())/d['EGOV_SAT_score'].std()
    # Continuous moderators are standardised; binary moderators stay 0/1
    m_series = d[mod]
    if m_series.nunique()>2 and m_series.dtype != object:
        d['m_c']=(m_series-m_series.mean())/m_series.std()
    else:
        d['m_c']=m_series.astype(float)
    d['inter'] = d['egov']*d['m_c']
    X = sm.add_constant(d[['egov','m_c','inter'] + [x for x in ['male','age','edu','hukou_urban'] if x in d.columns]])
    m = sm.OLS(d['GOV_TRUST_score'], X).fit()
    coef = m.params.get('inter'); p = m.pvalues.get('inter'); se=m.bse.get('inter')
    return {'moderator':label,'interaction_B':round(float(coef),3),'SE':round(float(se),3),
            'p':round(float(p),4),'signif':('p<.001' if p<.001 else 'p<.05' if p<.05 else 'n.s.'),
            'direction':('+ (strengthens)' if coef>0 else '- (weakens)'),'n':int(m.nobs)}

rows=[]
for mod,label in [('digaware','Digital awareness (nq8)'),('digtrain','Digital training will (nq10)'),
                  ('age_c','Age'),('edu_c','Education')]:
    try:
        rows.append(mod_model(mod, df, label))
    except Exception as e:
        rows.append({'moderator':label,'interaction_B':np.nan,'SE':np.nan,'p':np.nan,
                     'signif':f'ERR:{e}','direction':'','n':np.nan})
res = pd.DataFrame(rows)
res.to_csv(OUT / "moderation_results.csv", index=False)
print("\n=== Moderation results (EGOV_SAT x moderator -> GOV_TRUST) ===")
print(res[['moderator','interaction_B','SE','p','signif','direction','n']].to_string(index=False))

# Group-specific total associations by level of digital awareness
print("\n=== Total effect of EGOV on TRUST within each digital-awareness group ===")
for lvl in [1,2,3,4,5]:
    sub = df[df['digaware']==lvl]
    if len(sub)>50:
        X=sm.add_constant(sub[['EGOV_SAT_score']+controls].dropna())
        d=sub[['GOV_TRUST_score']+['EGOV_SAT_score']+controls].dropna()
        X=sm.add_constant(d[['EGOV_SAT_score']+controls]); m=sm.OLS(d['GOV_TRUST_score'],X).fit()
        print(f"  nq8={lvl} (n={len(d)}): β={m.params['EGOV_SAT_score']:.3f}, p={m.pvalues['EGOV_SAT_score']:.4f}")
```

S3.9 `14_lv_mediation.R`

```
#!/usr/bin/env Rscript
# Latent-variable mediation without a general factor (intermediate specification).
#
# Estimates EGOV -> PERF/DIG -> TRUST on item-level measures, separating the
# mediation estimate from measurement error. Because it does not model the shared
# evaluative factor, the bifactor model in 18_bifactor_sem_mediation.R is the
# specification reported in the manuscript; the path coefficients from this
# script are retained for transparency.
#
# Reads:  $DATA/item_level.csv, $DATA/analysis_dataset.csv
# Writes: $SUP/LV_mediation_results.csv, $OUT/three_factor_sem_fit.csv

# Paths are configurable so that the scripts run anywhere. Defaults assume the
# package layout: <package>/analysis_code, <package>/analysis_output,
# <package>/figures, and the restricted analytic datasets in <package>/data.
DATA <- Sys.getenv("CSGS_DATA_DIR", "data")
OUT <- Sys.getenv("CSGS_OUT_DIR", "analysis_output")
SUP <- Sys.getenv("CSGS_SUPERSEDED_DIR", "superseded_specifications")
FIG <- Sys.getenv("CSGS_FIG_DIR", "figures")
suppressMessages(library(lavaan))

rd <- read.csv(file.path(DATA, "item_level.csv"))
cols <- c(paste0("nq4_",1:5), paste0("hq4_",1:8), paste0("gq1_",1:5))
rd <- rd[complete.cases(rd[,cols]), ]
# Controls: item_level carries male and edu; age is aligned in from the analytic dataset
ds <- read.csv(file.path(DATA, "analysis_dataset.csv"))
rd$male <- ds$male; rd$age <- ds$age; rd$edu <- ds$edu
rd$hukou_urban <- ifelse(ds$hukou_type=="城镇户口",1,0)
rd$DIGATT <- ds$DIG_ATT_score
rd <- rd[complete.cases(rd), ]
cat("Sample:", nrow(rd), "\n")
model <- '
  EGOV =~ nq4_1+nq4_2+nq4_3+nq4_4+nq4_5
  PERF =~ hq4_1+hq4_2+hq4_3+hq4_4+hq4_5+hq4_6+hq4_7+hq4_8
  TRUST =~ gq1_1+gq1_2+gq1_3+gq1_4+gq1_5
  TRUST ~ c*EGOV + b1*PERF + b2*DIGATT + male + age + edu + hukou_urban
  PERF ~ a1*EGOV + male + age + edu + hukou_urban
  DIGATT ~ a2*EGOV + male + age + edu + hukou_urban
  ind1 := a1*b1
  ind2 := a2*b2
  total := c + a1*b1 + a2*b2
'
fit <- sem(model, data=rd, estimator="ML", se="bootstrap", bootstrap=1000)
cat("\n===== Latent-variable mediation paths (1000 bootstrap, BCa) =====\n")
p <- parameterEstimates(fit, boot.ci.type="bca.simple")
sel <- p[p$label %in% c("a1","a2","b1","b2","c","ind1","ind2","total"), c("label","est","se","ci.lower","ci.upper","pvalue")]
print(sel, row.names=FALSE)
write.csv(sel, file.path(SUP, "LV_mediation_results.csv"), row.names = FALSE)

# Fit of this three-factor structural model, quoted as a comparison in Section 4.4
fm <- fitMeasures(fit, c("chisq", "df", "cfi", "tli", "rmsea", "srmr"))
write.csv(data.frame(metric = names(fm), value = unname(fm)),
          file.path(OUT, "three_factor_sem_fit.csv"), row.names = FALSE)
cat("Saved ->", file.path(SUP, "LV_mediation_results.csv"),
    "and", file.path(OUT, "three_factor_sem_fit.csv"), "\n")
```

S3.10 `15_weighting.py`

```
#!/usr/bin/env python3
"""Post-stratification weighting diagnostic.

Supports the weighting robustness check reported in Section 4.6.

Constructs an approximate post-stratification weight from the sample's own
gender-by-age-group-by-urban-location cell frequencies (inverse cell frequency,
normalised to mean 1) and refits the main model with those weights. The weight
uses the sample's own cell distribution rather than external census margins, so
it is reported as an approximate check rather than a population correction.

Reads:  $DATA/analysis_dataset.csv
Writes: $OUT/weighting_comparison.csv
"""

import os
from pathlib import Path

# Paths are configurable so that the scripts run anywhere. Defaults assume the
# package layout: <package>/analysis_code, <package>/analysis_output,
# <package>/figures, and the restricted analytic datasets in <package>/data.
DATA = Path(os.environ.get("CSGS_DATA_DIR", "data"))
OUT = Path(os.environ.get("CSGS_OUT_DIR", "analysis_output"))
SUP = Path(os.environ.get("CSGS_SUPERSEDED_DIR", "superseded_specifications"))
FIG = Path(os.environ.get("CSGS_FIG_DIR", "figures"))
import numpy as np, pandas as pd
import statsmodels.api as sm
from pathlib import Path

ds = pd.read_csv(DATA / "analysis_dataset.csv")
df = pd.DataFrame(index=ds.index)
# Construct scores and controls come from the analytic dataset
df['EGOV'] = ds['EGOV_SAT_score']; df['PERF'] = ds['PERF_EVAL_score']
df['GOV_TRUST'] = ds['GOV_TRUST_score']; df['DIG'] = ds['DIG_ATT_score']
df['male'] = ds['male']; df['age'] = ds['age']; df['edu'] = ds['edu']
df['hukou_urban'] = ds['hukou_type'].map({'城镇户口':1,'农村户口':0})
controls=['male','age','edu','hukou_urban']

# --- Approximate post-stratification weight (gender x age group x urban) ---
# Age groups
bins=[0,24,34,44,54,64,200]; labels=['18-24','25-34','35-44','45-54','55-64','65+']
df['ageg'] = pd.cut(df['age'], bins=bins, labels=labels)
# Target margins: the sample's own cell distribution (approximate raking)
cell = df.groupby(['male','ageg','hukou_urban']).size().reset_index(name='freq')
df = df.merge(cell, on=['male','ageg','hukou_urban'], how='left')
df['w'] = (len(df) / df['freq'])  # inverse cell frequency
df['w'] = df['w'] / df['w'].mean()  # normalise to mean 1

def run(y, Xc, d, w=None, label=''):
    d = d[Xc + [y] + (['w'] if w else [])].dropna()
    X = sm.add_constant(d[Xc])
    if w is None:
        return sm.OLS(d[y], X).fit()
    return sm.WLS(d[y], X, weights=d[w]).fit()  # weights must actually be applied

m_un = run('GOV_TRUST', ['EGOV','PERF','DIG']+controls, df, label='unweighted')
m_wt = run('GOV_TRUST', ['EGOV','PERF','DIG']+controls, df, w='w', label='weighted')

print("=== Weighted estimation (approximate gender-age-urban raking) ===")
for name, m in [('Unweighted', m_un), ('Weighted', m_wt)]:
    eg=m.params.get('EGOV'); per=m.params.get('PERF'); dig=m.params.get('DIG')
    print(f"  {name}: EGOV β={eg:.3f} | PERF β={per:.3f} | DIG β={dig:.3f} | R²={m.rsquared_adj:.3f}")

# Change in the EGOV coefficient after weighting
print("\n[Result] change in the EGOV coefficient after weighting:",
      round(m_wt.params['EGOV'] - m_un.params['EGOV'], 3))
# Save
res=pd.DataFrame({
    'spec':['unweighted','weighted'],
    'EGOV_b':[m_un.params['EGOV'],m_wt.params['EGOV']],
    'PERF_b':[m_un.params['PERF'],m_wt.params['PERF']],
    'DIG_b':[m_un.params['DIG'],m_wt.params['DIG']],
    'adj_R2':[m_un.rsquared_adj,m_wt.rsquared_adj]})
res.to_csv(OUT / "weighting_comparison.csv", index=False)
print("\nSaved ->", OUT / "weighting_comparison.csv")
```

S3.11 `17_moderated_mediation.R`

```
#!/usr/bin/env Rscript
# Moderated mediation: does digital awareness condition the mediated and the
# direct pathways separately? Pre-computed product terms are used because the
# three-way interaction syntax is not available for this model.
#
# Supports the moderated-mediation results reported in Section 4.5.
#
# Reads:  $DATA/analysis_dataset.csv, $DATA/item_level.csv
# Writes: $OUT/moderated_mediation_results.csv

# Paths are configurable so that the scripts run anywhere. Defaults assume the
# package layout: <package>/analysis_code, <package>/analysis_output,
# <package>/figures, and the restricted analytic datasets in <package>/data.
DATA <- Sys.getenv("CSGS_DATA_DIR", "data")
OUT <- Sys.getenv("CSGS_OUT_DIR", "analysis_output")
SUP <- Sys.getenv("CSGS_SUPERSEDED_DIR", "superseded_specifications")
FIG <- Sys.getenv("CSGS_FIG_DIR", "figures")
suppressMessages(library(lavaan))
set.seed(42)  # fixed seed so the bootstrap results are reproducible

ds <- read.csv(file.path(DATA, "analysis_dataset.csv"))
rd <- read.csv(file.path(DATA, "item_level.csv"))
for (v in paste0("nq4_",1:5)) rd[[v]] <- as.numeric(rd[[v]])
for (v in paste0("hq4_",1:8)) rd[[v]] <- as.numeric(rd[[v]])
for (v in paste0("gq1_",1:5)) rd[[v]] <- as.numeric(rd[[v]])
rd$male <- ds$male; rd$age <- ds$age; rd$edu <- ds$edu
rd$hukou <- ifelse(ds$hukou_type=="城镇户口",1,0)
rd$DIG <- ds$DIG_ATT_score
# Digital awareness (nq8) coded 1-5 and centred
nq8_map <- c("完全不了解"=1,"比较不了解"=2,"一般"=3,"比较了解"=4,"非常了解"=5)
rd$digaware <- unname(nq8_map[as.character(ds$nq8)])
rd$digaware_c <- (rd$digaware - mean(rd$digaware, na.rm=TRUE))/sd(rd$digaware, na.rm=TRUE)
# Standardise the core constructs
rd$EG <- scale(rowMeans(rd[paste0("nq4_",1:5)]))[,1]
rd$PERF <- scale(rowMeans(rd[paste0("hq4_",1:8)]))[,1]
rd$TRUST <- scale(rowMeans(rd[paste0("gq1_",1:5)]))[,1]
# Pre-compute the product terms
rd$EGxDA <- rd$EG * rd$digaware_c
rd$PERFxDA <- rd$PERF * rd$digaware_c
rd <- rd[complete.cases(rd), ]
cat("Sample:", nrow(rd), "\n")

model <- '
  PERF ~ a0*EG + a1*digaware_c + a2*EGxDA + male + age + edu + hukou
  TRUST ~ c0*EG + b1*PERF + b2*PERFxDA + b3*digaware_c + c1*EGxDA + male + age + edu + hukou
  # Conditional indirect effect (a*b) at digital awareness -1SD / mean / +1SD
  ind_mean := a0*b1
  ind_low  := (a0 + a2*(-1))*(b1 + b2*(-1))
  ind_high := (a0 + a2*(1))*(b1 + b2*(1))
  # Moderated mediation: the high-minus-low difference
  mm_diff := ind_high - ind_low
'
fit <- sem(model, data=rd, estimator="ML", se="bootstrap", bootstrap=1000)
cat("\n===== Moderated mediation (digital awareness on the indirect and direct paths) =====\n")
p <- parameterEstimates(fit, boot.ci.type="bca.simple", level=.95)
sel <- p[p$label %in% c("a0","a2","b1","b2","c0","c1","ind_mean","ind_low","ind_high","mm_diff"),
c("label","est","ci.lower","ci.upper","pvalue")]
print(sel, row.names=FALSE)
write.csv(sel, file.path(OUT, "moderated_mediation_results.csv"), row.names = FALSE)
cat("Saved ->", file.path(OUT, "moderated_mediation_results.csv"), "\n")
```

S3.12 `18_bifactor_sem_mediation.R`

```
#!/usr/bin/env Rscript
# Main analysis: bifactor structural equation model with mediation.
#
# Decomposes the EGOV -> TRUST association into a direct path and two indirect
# paths while controlling a general evaluative factor G alongside the three
# specific factors. This is the specification reported in Section 4.4.
#
# Identification. In a bifactor model the general factor must be orthogonal to the
# specific factors: those covariances are fixed at zero below (G ~~ 0*...), and G
# therefore does not enter the structural equations. Without those constraints the
# model is not identified (lavaan reports that the information matrix cannot be
# inverted and the bootstrap becomes unstable).
#
# Reads:  $DATA/item_level.csv, $DATA/analysis_dataset.csv
# Writes: $OUT/bifactor_mediation_results.csv, $OUT/bifactor_sem_fit.csv,
#         $OUT/mediation_decomposition.csv

# Paths are configurable so that the scripts run anywhere. Defaults assume the
# package layout: <package>/analysis_code, <package>/analysis_output,
# <package>/figures, and the restricted analytic datasets in <package>/data.
DATA <- Sys.getenv("CSGS_DATA_DIR", "data")
OUT <- Sys.getenv("CSGS_OUT_DIR", "analysis_output")
SUP <- Sys.getenv("CSGS_SUPERSEDED_DIR", "superseded_specifications")
FIG <- Sys.getenv("CSGS_FIG_DIR", "figures")
suppressMessages(library(lavaan))
set.seed(42)  # fixed seed so the bootstrap results are reproducible

rd <- read.csv(file.path(DATA, "item_level.csv"))
for (v in c(paste0("nq4_", 1:5), paste0("hq4_", 1:8), paste0("gq1_", 1:5))) {
  rd[[v]] <- as.numeric(rd[[v]])
}
ds <- read.csv(file.path(DATA, "analysis_dataset.csv"))
rd$male <- ds$male; rd$age <- ds$age; rd$edu <- ds$edu
rd$hukou <- ifelse(ds$hukou_type == "城镇户口", 1, 0)
rd$DIG <- ds$DIG_ATT_score
rd <- rd[complete.cases(rd), ]
cat("Sample:", nrow(rd), "
")

model <- '
  G =~ nq4_1+nq4_2+nq4_3+nq4_4+nq4_5+hq4_1+hq4_2+hq4_3+hq4_4+hq4_5+hq4_6+hq4_7+hq4_8+gq1_1+gq1_2+gq1_3+gq1_4+gq1_5
  EGOV =~ nq4_1+nq4_2+nq4_3+nq4_4+nq4_5
  PERF =~ hq4_1+hq4_2+hq4_3+hq4_4+hq4_5+hq4_6+hq4_7+hq4_8
  TRUST =~ gq1_1+gq1_2+gq1_3+gq1_4+gq1_5
  G ~~ 0*EGOV + 0*PERF + 0*TRUST
  EGOV ~~ 0*PERF + 0*TRUST
  PERF ~~ 0*TRUST
  PERF ~ a1*EGOV + male + age + edu + hukou
  DIG ~ a2*EGOV + male + age + edu + hukou
  TRUST ~ c*EGOV + b1*PERF + b2*DIG + male + age + edu + hukou
  ind1 := a1*b1
  ind2 := a2*b2
  total := c + a1*b1 + a2*b2
'
fit <- sem(model, data = rd, estimator = "ML", std.lv = TRUE)
fm <- fitMeasures(fit, c("chisq", "df", "cfi", "tli", "rmsea", "srmr"))
cat("=== Bifactor SEM fit ===
"); print(round(fm, 3))

fitb <- sem(model, data = rd, estimator = "ML", std.lv = TRUE,
            se = "bootstrap", bootstrap = 1000)
nfail <- sum(!lavInspect(fitb, "boot"))
cat("bootstrap runs that did not converge:", nfail, "of 1000
")
if (nfail > 0) warning("some bootstrap runs failed; the confidence intervals may be unreliable")

lab <- c("a1", "a2", "b1", "b2", "c", "ind1", "ind2", "total")
p <- parameterEstimates(fitb, boot.ci.type = "bca.simple", level = .95)
raw <- p[p$label %in% lab, c("label", "est", "ci.lower", "ci.upper", "pvalue")]

# Standardised estimates, which is what the manuscript reports as beta
st <- standardizedSolution(fitb, boot.ci.type = "bca.simple", level = .95)
key <- ifelse(st$op == ":=", st$lhs, st$label)
keep <- key %in% lab
std <- data.frame(label = key[keep], est.std = st$est.std[keep],
                  std.ci.lower = st$ci.lower[keep], std.ci.upper = st$ci.upper[keep])
out <- merge(raw, std, by = "label", sort = FALSE)
out <- out[match(lab, out$label), ]
print(out, row.names = FALSE)
write.csv(out, file.path(OUT, "bifactor_mediation_results.csv"), row.names = FALSE)
write.csv(data.frame(metric = names(fm), value = unname(fm)),
          file.path(OUT, "bifactor_sem_fit.csv"), row.names = FALSE)

# Decomposition of the total association, computed from the standardised estimates
# so that the shares can be checked by hand against the reported coefficients.
gv <- function(l, col) out[[col]][out$label == l]
```

```
r3 <- function(x) sprintf("%.3f", round(x, 3))
pct <- function(x, tot) sprintf("%.1f", round(100 * x / tot, 1))
tot <- gv("total", "est.std")
dec <- data.frame(
  effect = c("Total", "Direct (c)", "Indirect via PERF", "Indirect via DIG"),
  est = c(r3(tot), r3(gv("c", "est.std")), r3(gv("ind1", "est.std")), r3(gv("ind2", "est.std"))),
  pct = c("100.0",
          pct(round(gv("c", "est.std"), 3), round(tot, 3)),
          pct(round(gv("ind1", "est.std"), 3), round(tot, 3)),
          pct(round(gv("ind2", "est.std"), 3), round(tot, 3))))
write.csv(dec, file.path(OUT, "mediation_decomposition.csv"), row.names = FALSE)
print(dec)
cat("Saved ->", file.path(OUT, "bifactor_mediation_results.csv"), "and",
    file.path(OUT, "bifactor_sem_fit.csv"), "and",
    file.path(OUT, "mediation_decomposition.csv"), "
")
```

S3.13 `19_dig_reliability_sensitivity.R`

```
#!/usr/bin/env Rscript
# Sensitivity of the belief pathway to the reliability of the single-item mediator.
#
# Digital governance belief is measured by one item. This script re-estimates the
# main model treating that item as a latent variable whose residual variance is
# fixed to (1 - reliability) * observed variance, for reliabilities of .80, .70 and
# .60, and for the error-free case. It reports how far measurement error attenuates
# the belief pathway. Supports the sensitivity result reported in Section 4.4.
#
# Reads:  $DATA/item_level.csv, $DATA/analysis_dataset.csv
# Writes: $OUT/dig_reliability_sensitivity.csv

# Paths are configurable so that the scripts run anywhere. Defaults assume the
# package layout: <package>/analysis_code, <package>/analysis_output,
# <package>/figures, and the restricted analytic datasets in <package>/data.
DATA <- Sys.getenv("CSGS_DATA_DIR", "data")
OUT <- Sys.getenv("CSGS_OUT_DIR", "analysis_output")
SUP <- Sys.getenv("CSGS_SUPERSEDED_DIR", "superseded_specifications")
FIG <- Sys.getenv("CSGS_FIG_DIR", "figures")
suppressMessages(library(lavaan))
set.seed(42)

rd <- read.csv(file.path(DATA, "item_level.csv"))
for (v in c(paste0("nq4_", 1:5), paste0("hq4_", 1:8), paste0("gq1_", 1:5))) {
  rd[[v]] <- as.numeric(rd[[v]])
}
ds <- read.csv(file.path(DATA, "analysis_dataset.csv"))
rd$male <- ds$male; rd$age <- ds$age; rd$edu <- ds$edu
rd$hukou <- ifelse(ds$hukou_type == "城镇户口", 1, 0)
rd$DIG <- ds$DIG_ATT_score
rd <- rd[complete.cases(rd), ]

measurement <- '
  G =~ nq4_1+nq4_2+nq4_3+nq4_4+nq4_5+hq4_1+hq4_2+hq4_3+hq4_4+hq4_5+hq4_6+hq4_7+hq4_8+gq1_1+gq1_2+gq1_3+gq1_4+gq1_5
  EGOV =~ nq4_1+nq4_2+nq4_3+nq4_4+nq4_5
  PERF =~ hq4_1+hq4_2+hq4_3+hq4_4+hq4_5+hq4_6+hq4_7+hq4_8
  TRUST =~ gq1_1+gq1_2+gq1_3+gq1_4+gq1_5
  G ~~ 0*EGOV + 0*PERF + 0*TRUST
  EGOV ~~ 0*PERF + 0*TRUST
  PERF ~~ 0*TRUST'

observed_var <- var(rd$DIG)
rows <- list()
for (rho in c(1, 0.8, 0.7, 0.6)) {
  structural <- if (rho >= 0.999) {
    'PERF ~ a1*EGOV + male+age+edu+hukou
     DIG ~ a2*EGOV + male+age+edu+hukou
     TRUST ~ c*EGOV + b1*PERF + b2*DIG + male+age+edu+hukou'
  } else {
    paste0('DIG_lat =~ 1*DIG
     DIG ~~ ', format((1 - rho) * observed_var, digits = 10), '*DIG
     PERF ~ a1*EGOV + male+age+edu+hukou
     DIG_lat ~ a2*EGOV + male+age+edu+hukou
     TRUST ~ c*EGOV + b1*PERF + b2*DIG_lat + male+age+edu+hukou')
  }
  model <- paste(measurement, structural,
                 'ind1 := a1*b1
  ind2 := a2*b2
  total := c + a1*b1 + a2*b2', sep = "
")
  fit <- sem(model, data = rd, estimator = "ML")
  st <- standardizedSolution(fit)
  pe <- parameterEstimates(fit)
  g <- function(l) st$est.std[st$label == l]
  rows[[length(rows) + 1]] <- data.frame(
    reliability = rho,
    a2 = round(g("a2"), 3), b2 = round(g("b2"), 3),
    indirect_DIG = round(g("ind2"), 4),
    ind2_p = round(pe$pvalue[pe$label == "ind2"], 4),
    total = round(g("total"), 3))
  cat(sprintf("reliability %.1f: indirect via DIG = %.4f (p = %.3f)
",
              rho, g("ind2"), pe$pvalue[pe$label == "ind2"]))
}
out <- do.call(rbind, rows)
write.csv(out, file.path(OUT, "dig_reliability_sensitivity.csv"), row.names = FALSE)
cat("Saved ->", file.path(OUT, "dig_reliability_sensitivity.csv"), "
")
```

S3.14 `11_figure2.py`

```
#!/usr/bin/env python3
"""Figure 2: decomposition of the satisfaction–trust association.

Bar chart of the total association and its direct, perceived-performance, and
digital-governance-belief components. All values are read from the analysis
output, so the figure cannot drift from the reported numbers.

Reads:  $OUT/mediation_decomposition.csv, $OUT/bifactor_mediation_results.csv
Writes: $FIG/Figure2_mediation.png, $FIG/Figure2_mediation.pdf
"""

import os
from pathlib import Path

# Paths are configurable so that the scripts run anywhere. Defaults assume the
# package layout: <package>/analysis_code, <package>/analysis_output,
# <package>/figures, and the restricted analytic datasets in <package>/data.
DATA = Path(os.environ.get("CSGS_DATA_DIR", "data"))
OUT = Path(os.environ.get("CSGS_OUT_DIR", "analysis_output"))
SUP = Path(os.environ.get("CSGS_SUPERSEDED_DIR", "superseded_specifications"))
FIG = Path(os.environ.get("CSGS_FIG_DIR", "figures"))
import matplotlib
matplotlib.use('Agg')
matplotlib.rcParams['font.family'] = 'Arial'
matplotlib.rcParams['svg.fonttype'] = 'none'  # keep text editable
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd

dec = pd.read_csv(OUT / "mediation_decomposition.csv").set_index("effect")
est = pd.read_csv(OUT / "bifactor_mediation_results.csv").set_index("label")

order = ["Total", "Direct (c)", "Indirect via PERF", "Indirect via DIG"]
labels = ['Total\neffect', 'Direct\n(c\u2032)', 'Indirect\nvia PERF', 'Indirect\nvia DIG_ATT']
values = [float(dec.loc[k, "est"]) for k in order]  # standardised estimates
pct = [f'{float(dec.loc[k, "pct"]):.1f}%' for k in order]
# Okabe-Ito colour-blind-safe palette
colors = ['#0072B2', '#D55E00', '#009E73', '#CC79A7']

fig, ax = plt.subplots(figsize=(6.5, 4.5), dpi=300)
y = np.arange(len(labels))[::-1]
ax.barh(y, values, color=colors, height=0.62, edgecolor='white')

for yi, v, p in zip(y, values, pct):
    ax.text(v + 0.012, yi, f'\u03b2 = {v:.3f}\n({p})', va='center', fontsize=10, color='#333')

ax.set_yticks(y)
ax.set_yticklabels(labels, fontsize=11)
ax.set_xlabel('Association with government trust (\u03b2)', fontsize=12)
ax.set_xlim(0, 0.60)
ax.set_title('Decomposition of the e-government satisfaction\u2013government trust association', fontsize=11)
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)
ax.axvline(values[0], color='grey', ls='--', lw=0.8, alpha=0.6)

plt.tight_layout()
FIG.mkdir(parents=True, exist_ok=True)
out = FIG / "Figure2_mediation"
plt.savefig(str(out) + '.png', dpi=600, bbox_inches='tight')
plt.savefig(str(out) + '.pdf', bbox_inches='tight')
print('Figure 2 written:', out, '(png/pdf)')
print('values:', dict(zip(order, np.round(values, 3))))
```

S3.15 `12_figure3.py`

```
#!/usr/bin/env python3
"""Figure 3: satisfaction-trust association by level of digital awareness.

Split-sample regressions of government trust on e-government satisfaction
within each of the five levels of digital awareness, with 95 per cent
confidence intervals. Also writes the group descriptives and the plotted
estimates as machine-readable tables.

Reads:  $DATA/analysis_dataset.csv
Writes: $FIG/Figure3_gradient.png, $FIG/Figure3_gradient.pdf,
        $OUT/group_trust_distribution.csv, $OUT/figure3_slopes.csv
"""

import os
from pathlib import Path

# Paths are configurable so that the scripts run anywhere. Defaults assume the
# package layout: <package>/analysis_code, <package>/analysis_output,
# <package>/figures, and the restricted analytic datasets in <package>/data.
DATA = Path(os.environ.get("CSGS_DATA_DIR", "data"))
OUT = Path(os.environ.get("CSGS_OUT_DIR", "analysis_output"))
SUP = Path(os.environ.get("CSGS_SUPERSEDED_DIR", "superseded_specifications"))
FIG = Path(os.environ.get("CSGS_FIG_DIR", "figures"))
import matplotlib
matplotlib.use('Agg')
matplotlib.rcParams['font.family'] = 'Arial'
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import statsmodels.api as sm

df = pd.read_csv(DATA / "analysis_dataset.csv")
df['hukou_urban'] = df['hukou_type'].map({'\u57ce\u9547\u6237\u53e3': 1, '\u519c\u6751\u6237\u53e3': 0})
AWARE = {'\u5b8c\u5168\u4e0d\u4e86\u89e3': 1, '\u6bd4\u8f83\u4e0d\u4e86\u89e3': 2,
         '\u4e00\u822c': 3, '\u6bd4\u8f83\u4e86\u89e3': 4, '\u975e\u5e38\u4e86\u89e3': 5}
df['nq8_num'] = df['nq8'].map(AWARE)
controls = ['male', 'age', 'edu', 'hukou_urban']

labels = {1: 'Not at all\nfamiliar', 2: 'Not very\nfamiliar', 3: 'Neutral',
          4: 'Somewhat\nfamiliar', 5: 'Very\nfamiliar'}
betas, ses, ns, ci_lo, ci_hi, ps = [], [], [], [], [], []
for lvl in [1, 2, 3, 4, 5]:
    sub = df[df['nq8_num'] == lvl]
    d = sub[['GOV_TRUST_score', 'EGOV_SAT_score'] + controls].dropna()
    X = sm.add_constant(d[['EGOV_SAT_score'] + controls])
    m = sm.OLS(d['GOV_TRUST_score'], X).fit()
    b, se = m.params['EGOV_SAT_score'], m.bse['EGOV_SAT_score']
    betas.append(b); ses.append(se); ns.append(len(d)); ps.append(float(m.pvalues['EGOV_SAT_score']))
    ci_lo.append(b - 1.96 * se); ci_hi.append(b + 1.96 * se)

betas = np.array(betas); ses = np.array(ses)
xs = np.arange(1, 6)

# --- Machine-readable tables backing the figure ---
slopes = pd.DataFrame({
    'digital_awareness': [1, 2, 3, 4, 5], 'n': ns,
    'b': np.round(betas, 3), 'se': np.round(ses, 3),
    'ci_lower': np.round(ci_lo, 3), 'ci_upper': np.round(ci_hi, 3),
    'p': np.round(ps, 4)})
slopes.to_csv(OUT / "figure3_slopes.csv", index=False)

groups = (df.dropna(subset=['nq8_num'])
            .groupby('nq8_num')
            .agg(n=('GOV_TRUST_score', 'size'),
                 trust_m=('GOV_TRUST_score', 'mean'), trust_sd=('GOV_TRUST_score', 'std'),
                 egov_m=('EGOV_SAT_score', 'mean'), egov_sd=('EGOV_SAT_score', 'std'))
            .reset_index().rename(columns={'nq8_num': 'digaware_level'}))
for c in ['trust_m', 'trust_sd', 'egov_m', 'egov_sd']:
    groups[c] = groups[c].round(3)
groups['digaware_level'] = groups['digaware_level'].astype(int)
groups.to_csv(OUT / "group_trust_distribution.csv", index=False)

# --- Figure ---
fig, ax = plt.subplots(figsize=(7, 5), dpi=300)
ax.errorbar(xs, betas, yerr=1.96 * ses, fmt='o', color='#4472C4', ecolor='#999',
            elinewidth=2, capsize=4, markersize=9, zorder=3,
            label='Point estimate \u00b1 95% CI')
ax.plot(xs, betas, color='#2E4B8F', lw=2, alpha=0.6, zorder=2)
for x, b, se, pv in zip(xs, betas, ses, ps):
    star = '***' if pv < .001 else ('**' if pv < .01 else ('*' if pv < .05 else ''))
    ax.annotate(f'b = {b:.3f}{star}', (x, b), textcoords='offset points',
                xytext=(0, 10), ha='center', fontsize=9, color='#333')
ax.axhline(0, color='#ccc', lw=0.8, ls='--')
ax.set_xlabel('Digital awareness (1-5)', fontsize=12)
ax.set_ylabel('Association with government trust (b)', fontsize=12)
ax.set_xticks(xs)
```

```
ax.set_xticklabels([labels[i] for i in xs], fontsize=9)
ax.set_ylim(0.15, 0.65)
ax.set_title('E-gov satisfaction\u2013government trust association by digital awareness', fontsize=11)
ax.legend(loc='lower left', fontsize=9)
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)

plt.tight_layout()
FIG.mkdir(parents=True, exist_ok=True)
out = FIG / "Figure3_gradient.png"
plt.savefig(out, dpi=600, bbox_inches='tight')
plt.savefig(str(out).replace('.png', '.pdf'), bbox_inches='tight')
print("Figure 3 written:", out, "(png/pdf)")
print("betas:", dict(zip(range(1, 6), np.round(betas, 3))))
```